SAMIRA SOUIT

ORQUESTRAÇÃO DE SISTEMAS PRODUTIVOS DISPERSOS

São Paulo
2013

SAMIRA SOUIT

ORQUESTRAÇÃO DE SISTEMAS PRODUTIVOS DISPERSOS

Dissertação apresentada à Escola Politécnica
da Universidade de São Paulo para obtenção
do título de Mestre em Ciências.

Área de Concentração:

Engenharia de Controle e Automação
Mecânica.

Orientador:

Prof. Dr. Paulo Eigi Miyagi

São Paulo
2013

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 07 de julho de 2013.

Assinatura do autor

Assinatura do orientador

FICHA CATALOGRÁFICA

Souit, Samira

**Orquestração de sistemas produtivos dispersos / S. Souit. --
São Paulo, 2013.
p.119**

**Dissertação (Mestrado) - Escola Politécnica da Universidade
de São Paulo. Departamento de Engenharia Mecatrônica e de
Sistemas Mecânicos.**

**1. Sistemas de manufatura 2. Sistemas distribuídos 3. World
Wide Web 4. Serviços 5. Controle de processos I. Universidade
de São Paulo. Escola Politécnica. Departamento de Engenharia
Mecatrônica e de Sistemas Mecânicos II. t.**

Dedico este trabalho à minha mãe Suely,
e ao meu pai, Emir.

AGRADECIMENTOS

Sinceros agradecimentos a todos que de alguma forma contribuíram para este projeto.

Agradeço a minha mãe, Suely, a quem devo e dedico tudo, sem seu amor e a sua dedicação incondicional, jamais conseguiria. Ao meu pai, Emir, por me apoiar e me fazer acreditar em mim mesma quando os problemas pareciam sem solução, sua paixão pela engenharia sempre me inspirou. Agradeço à minha avó, Odete, por ser meu exemplo, me ensinar a ter coragem e a sempre enfrentar os problemas sem medo. À minha irmã, Camila, e a toda a minha família, por serem minha estrutura, vocês são essenciais para mim.

Eterna gratidão ao Norman, por toda a ajuda neste projeto, por me apoiar, acalmar, entender e estar ao meu lado nesta fase difícil, sua contribuição foi fundamental.

Agradeço ao Prof. Paulo Eigi Miyagi, por toda sua orientação, paciência e por sua exigência, que sempre me fez conseguir ir além do que esperava.

A todos os colegas e amigos da Poli, em especial ao Caio, ao Prof. Fabricio, ao Marcosiris, ao Prof. Diolino, ao José Isidro e a todos os outros por me ajudarem e me orientarem em diversos momentos. Aos colegas da ABB, pelo apoio durante o período.

Ao Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos da Escola Politécnica da USP, ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), que institucionalmente viabilizaram este trabalho.

RESUMO

De acordo com as tendências de demanda por produtos cada vez mais personalizados com menor *time-to-market* possível, evidencia-se a necessidade da composição de sistemas produtivos (SPs) com uma estrutura reconfigurável para manter a competitividade frente aos crescentes desafios de um mercado globalizado. Esta capacidade de reconfiguração envolve não só as estruturas internas dos SPs, mas também a participação destes na execução de forma integrada e coordenada em um processo produtivo mais complexo. Dessa maneira, esta dissertação introduz uma arquitetura de supervisão e controle que integra e coordena um conjunto de SPs fisicamente dispersos, cujas funcionalidades estão disponibilizadas como serviços para a execução dos processos produtivos e que, para a supervisão e controle desses SPs dispersos adota-se a técnica de “orquestração” junto com o conceito de SOA (*Service-Oriented Architecture*) de modo que em caso de indisponibilidade de algum SP, ocorra a seleção de outro SP de acordo com um critério de prioridade que melhor atende aos requisitos para a manufatura dos produtos. Desta forma tem-se uma solução para a inclusão dinâmica de SPs no sistema. Como o SP disperso pode ser caracterizado como um sistema a eventos discretos, as funcionalidades das partes que compõem a arquitetura proposta é desenvolvida segundo uma metodologia de projeto baseada em modelos em rede de Petri cujas propriedades são exploradas para a avaliação desta arquitetura. O texto apresenta ainda um exemplo ilustrativo das principais funcionalidades da arquitetura proposta implementada para um SP disperso.

ABSTRACT

According to trends in demand for customized products with less time-to-market possible, it is noticeable the need for a composition of productive systems (PSs) with a reconfigurable structure to maintain competitiveness against the increasing challenges of globalized market. This ability to reconfigure involves not only the internal structure of the PS, but also their participation implementing an integrated and coordinated environment in a more complex productive process. Thus, this work presents a supervisory control architecture that integrates and coordinates a set of physically dispersed PSs, whose functionalities are made available as services for the execution of productive processes and, for the supervision and control of dispersed PSs it is adopted the "orchestration" technique, through the approach of SOA (Service-oriented architecture), that allows, in case of unavailability of any PS, the selection of another PS according to priority criteria that best meet the requirements for the manufacture of products. This is a solution for dynamic inclusion of PSs into the system. Since the dispersed PS can be characterized as a discrete event system, the functionalities of the parts that compose the proposed architecture is developed according to a design methodology based on Petri Net models, whose properties are explored to evaluate the architecture. Also is presented an illustrative example of the main functionalities of the proposed architecture implemented for dispersed SPs.

LISTA DE FIGURAS

FIGURA 1.1 - ESTRUTURA TÍPICA DE CAMADAS DE SPs. ADAPTADO DE (GROBA ET AL., 2008).....	4
FIGURA 1.2 - EXEMPLO DE UM <i>MIDDLEWARE</i> INTEGRANDO CAMADAS DE UM SP. ADAPTADO DE (GROBA ET AL., 2008)	6
FIGURA 1.3 – ILUSTRAÇÃO DE ORQUESTRAÇÃO DE SPs EXPOSTOS COMO SERVIÇO.....	8
FIGURA 2.1 - COMPONENTES DO WEB SERVICE. ADAPTADO DE (GOTTSCHALK, 2000)	13
FIGURA 2.2 - PROTOCOLOS DE SERVIÇOS PARA A IMPLEMENTAÇÃO DE WS (GARCIA MELO, 2011).	15
FIGURA 2.3 - ESTRUTURA DOS DOCUMENTOS WSDL 1.0 E WSDL 2.0 (MINOR, 2012)	17
FIGURA 2.4 – ARQUITETURA DE UM WS COM SUPORTE A COMPOSIÇÃO DINÂMICA (MARTINS, 2007)	19
FIGURA 2.5 - A) EXEMPLO DE MODELO DE FUSÃO POR <i>TRANSIÇÕES</i> , B) MODELO SIMPLIFICADO (GARCIA MELO, 2011).....	35
FIGURA 2.6 – COMPONENTES DO PFS. (GARCIA MELO, 2011)	37
FIGURA 3.1 - ARQUITETURA PROPOSTA EM (GARCIA MELO, 2011)	42
FIGURA 3.2 - ARQUITETURA PROPOSTA NESTE TRABALHO, COM A RECONFIGURAÇÃO DO NÍVEL 2	42
FIGURA 3.3 - ARQUITETURA PARA ORQUESTRAÇÃO DE SPs DISPERSOS	43
FIGURA 3.4 - PROCESSO PRODUTIVO GLOBAL	44
FIGURA 3.5 - DIAGRAMA DE CASOS DE USO DO " <i>SERVIÇO DE COORDENAÇÃO</i> "	48
FIGURA 3.6 - DIAGRAMA DE CASOS DE USO DO " <i>SERVIÇO DE INTEGRAÇÃO E DESCOBERTA</i> "	49
FIGURA 4.1 - REFINAMENTO DAS ATIVIDADES [<i>CADASTRO DE SPs</i>] E [<i>FUNÇÕES DE CADASTRO SP</i>].....	51
FIGURA 4.2 – MODELO EM RDP DAS ATIVIDADES [<i>CADASTRO DE SPs</i>] E [<i>FUNÇÕES DE CADASTRO SP</i>]	52
FIGURA 4.3 - REFINAMENTO DO MODELO EM PFS DAS ATIVIDADES [<i>ATUALIZA ESTADOS</i>] E [<i>FUNÇÕES DE ATUALIZAÇÃO DE ESTADOS</i>]	55
FIGURA 4.4 – MODELO EM RDP DAS ATIVIDADES [<i>ATUALIZA ESTADOS</i>] E [<i>FUNÇÕES DE ATUALIZAÇÃO DE ESTADO DO SP</i>]	56
FIGURA 4.5 - REFINAMENTO DO MODELO EM PFS PARA AS ATIVIDADES [<i>SOLICITA SPs</i>] E [<i>FUNÇÕES DE SOLICITAÇÃO DE SPs</i>].....	59
FIGURA 4.6 - MODELO EM RDP DAS ATIVIDADES [<i>SOLICITA SPs</i>] E [<i>FUNÇÕES DE SOLICITAÇÃO DE SPs</i>]	60
FIGURA 4.7 - REFINAMENTO DO MODELO EM PFS DAS ATIVIDADES [<i>SOLICITA ÚNICO SP</i>] E [<i>FUNÇÕES DE SOLICITAÇÃO DE ÚNICO SP</i>]	63
FIGURA 4.8 - MODELO EM RDP DAS ATIVIDADES [<i>SOLICITA ÚNICO SP</i>] E [<i>FUNÇÕES DE SOLICITAÇÃO DE ÚNICO SP</i>]	64
FIGURA 4.9 - REFINAMENTO EM PFS DAS ATIVIDADES [<i>NOVO PEDIDO</i>] E [<i>FUNÇÕES DE REGISTRO DE PEDIDO</i>]	67
FIGURA 4.10 - REFINAMENTO EM PFS DAS ATIVIDADES [<i>SELECIONA CRITÉRIO</i>] E [<i>FUNÇÕES DE REGISTRO DE CRITÉRIO PARA PEDIDO</i>]	67
FIGURA 4.11 - MODELO EM RDP DAS ATIVIDADES [<i>NOVO PEDIDO</i>] E [<i>FUNÇÕES DE REGISTRO DE PEDIDO</i>]	68
FIGURA 4.12 – MODELO EM RDP DAS ATIVIDADES [<i>SELECIONA CRITÉRIO</i>] E [<i>FUNÇÕES DE REGISTRO DE CRITÉRIO PARA PEDIDO</i>]	71
FIGURA 4.13 - REFINAMENTO EM PFS DA ATIVIDADE [<i>CONSULTA PEDIDOS</i>].....	73
FIGURA 4.14 – MODELO FUNCIONAL EM RDP DA ATIVIDADE [<i>CONSULTA PEDIDOS</i>]	74
FIGURA 4.15 – REFINAMENTO EM PFS DA ATIVIDADE [<i>DISPONIBILIDADE DE SPs</i>]	76
FIGURA 4.16 – MODELO FUNCIONAL EM RDP PARA A ATIVIDADE [<i>DISPONIBILIDADE DE SPs</i>]	77
FIGURA 4.17 - REFINAMENTO EM PFS DA ATIVIDADE [<i>EXECUÇÃO ETAPA SPx</i>].....	79
FIGURA 4.18 - MODELO FUNCIONAL EM RDP DA ATIVIDADE [<i>EXECUÇÃO ETAPA SPx</i>]	79

FIGURA 4.19 - REFINAMENTO EM PFS DA ATIVIDADE [<i>RESPOSTA SERVIÇO SPx</i>]	82
FIGURA 4.20 - MODELO FUNCIONAL EM RDP DA ATIVIDADE [<i>RESPOSTA SERVIÇO SPx</i>]	82
FIGURA 4.21 - REFINAMENTO EM PFS DA ATIVIDADE [<i>REGISTRA HISTÓRICO PEDIDO</i>]	84
FIGURA 4.22 - MODELO FUNCIONAL EM RDP DA ATIVIDADE [<i>REGISTRA HISTÓRICO PEDIDO</i>]	84
FIGURA 5.1 - FOTO ILUSTRATIVA DO SISTEMA FLEXÍVEL E AUTOMATIZADO DE "ALIMENTAÇÃO", "INSPEÇÃO", "TRANSPORTE" E "MONTAGEM" QUE EMULAM SPs DISPERSOS	86
FIGURA 5.2 - PEÇAS A SEREM MONTADAS: CORPO, PINO, MOLA, TAMPA	87
FIGURA 5.3 - INTERFACE PARA O <i>INTEGRADOR DE SPs</i>	88
FIGURA 5.4 - CÓDIGO EM VB.NET COM EXEMPLO DE IMPLEMENTAÇÃO DO MÉTODO "COMPARADOR_WSDL"	91
FIGURA 5.5 - EXEMPLO DE IMPLEMENTAÇÃO DA INTERFACE DO <i>CLIENTE</i>	92

LISTA DE TABELAS

TABELA 2.1 - SÍNTESE DE ASPECTOS RELEVANTES DOS TRABALHOS DE SOA APLICADOS A ÁREA DE SPs. ADAPTADO DE (GARCIA MELO, 2011).....	28
TABELA 4.1 - DESCRIÇÃO DOS ELEMENTOS DO MODELO DE RdP PARA A ATIVIDADE [CADASTRO DE SP]	53
TABELA 4.2 - DESCRIÇÃO DOS ELEMENTOS DO MODELO DE RdP PARA ATIVIDADE[FUNÇÕES DE CADASTRO DE SP]	53
TABELA 4.3 - DESCRIÇÃO DOS ELEMENTOS DO MODELO DE RdP PARA A ATIVIDADE [ATUALIZA ESTADOS]	57
TABELA 4.4 - DESCRIÇÃO DOS ELEMENTOS DO MODELO DE RdP PARA A ATIVIDADE [FUNÇÕES DE ATUALIZAÇÃO DE ESTADO DO SP]	57
TABELA 4.5 – DESCRIÇÃO DOS ELEMENTOS DO MODELO DE RdP PARA AS ATIVIDADES [SOLICITA SPs]	60
TABELA 4.6 - DESCRIÇÃO DOS ELEMENTOS DO MODELO DE RdP PARA A ATIVIDADE [FUNÇÕES DE SOLICITAÇÃO DE SPs]	62
TABELA 4.7 – DESCRIÇÃO DOS ELEMENTOS DO MODELO DE RdP PARA AS ATIVIDADES [SOLICITA ÚNICO SP]	64
TABELA 4.8 - DESCRIÇÃO DOS ELEMENTOS DO MODELO DE RdP PARA AS ATIVIDADES [FUNÇÕES DE SOLICITAÇÃO DE ÚNICO SP]	65
TABELA 4.9 – DESCRIÇÃO DOS ELEMENTOS EM RdP DAS ATIVIDADES [NOVO PEDIDO]	69
TABELA 4.10 - DESCRIÇÃO DOS ELEMENTOS EM RdP DA ATIVIDADE [FUNÇÕES DE REGISTRO DE PEDIDO]	70
TABELA 4.11 – DESCRIÇÃO DOS ELEMENTOS DO MODELO EM RdP DAS ATIVIDADES [SELECIONA CRITÉRIO] E [FUNÇÕES DE REGISTRO DE PEDIDO]	71
TABELA 4.12 - DESCRIÇÃO DOS ELEMENTOS DO MODELO EM RdP DA ATIVIDADE [FUNÇÕES DE REGISTRO DE CRITÉRIO PARA PEDIDO]	72
TABELA 4.13 – DESCRIÇÃO DOS ELEMENTOS DO MODELO EM RdP DA ATIVIDADE [CONSULTA PEDIDOS]	74
TABELA 4.14 - TABELA COM ESTADOS DOS PEDIDOS	76
TABELA 4.15 – DESCRIÇÃO DOS ELEMENTOS EM RdP PARA A ATIVIDADE [DISPONIBILIDADE DE SPs]	77
TABELA 4.16 – DESCRIÇÃO DOS ELEMENTOS EM RdP PARA A ATIVIDADE [EXECUÇÃO ETAPA SPx]	80
TABELA 4.17 – DESCRIÇÃO DOS ELEMENTOS EM RdP PARA A ATIVIDADE [RESPOSTA SERVIÇO SPx]	82
TABELA 4.18 - DESCRIÇÃO DOS ELEMENTOS EM RdP DA ATIVIDADE [REGISTRA HISTÓRICO PEDIDO]	84

LISTA DE SIGLAS

Sigla	Descrição
AWS	<i>Agent-based Web Services</i>
BEEP	<i>Blocks Extensible Exchange Protocol</i>
BPEL	<i>Business Protocol Executable Language</i>
BPML	<i>Business Process Modeling Language</i>
CNC	Controle Numérico Computadorizado
DAML-S	<i>DARPA Agent Mark-up Language for Services</i>
DCS	<i>Distributed Control Systems</i>
DHCP	<i>Dynamic Host Configuration Protocol</i>
DMSPM	<i>Dynamic Manufacturing Service Provision Mechanism</i>
DPWS	<i>Devices Profile for Web Services</i>
EPUSP	Escola Politécnica da Universidade de São Paulo
FTP	<i>File Transfer Protocol</i>
HTTP	<i>Hyper Text Transfer Protocol</i>
LAN	Rede de Comunicação Local
LSA	Laboratório de Sistemas de Automação
MeC	<i>Mechatronic Components</i>
MMS	<i>Manufacturing Message Specification</i>
NC	Comando Numérico
OPC	<i>Object Linking and Embedding for Process Control</i>
OWL-S	<i>Ontology Web Language for services</i>
PCC	<i>Process Control Components</i>
PFS	<i>Production Flow Schema</i>
PFS/RdP	<i>Production Flow Schema/Rede de Petri</i>
PMCs	<i>Process Module Controller</i>
QoS	<i>Quality of Services</i>
RdP	Rede de Petri
RPC	<i>Remote Procedure Call</i>
SCADA	<i>Supervisory Control and Data Acquisition</i>
SCTSP	Sistema Coordenado de Teleoperação de sistemas produtivos
SED	Sistema a Eventos Discretos
SM	Sistema de Manufatura
SMTP	<i>Simple Mail Transfer Protocol</i>
SOA	<i>Service-oriented Architecture</i>
SOAP	<i>Simple Object Access Protocol</i>
SOMG	<i>Service Oriented Manufacturing Grid</i>
SP	Sistema Produtivo
TM	<i>Transfer Module</i>
UDP	<i>User Datagram Protocol</i>
UML	<i>Unified Modeling Language</i>
URL	<i>Uniform Resource Locator</i>
USDL	<i>Unified Service Description Language</i>
XML	<i>Extensible Markup Language</i>
XSD	<i>XML Schema Definition</i>
WS	<i>Web Service</i>
WSBPEL	<i>Process Execution Language for Web Services</i>
WSCI	<i>Web Service Choreography Interface</i>
WSCPC	<i>Web Services for Collaborative Product Commerce</i>
WSDL	<i>Web Service Description Language</i>
WSFL	<i>Web Services Flow Language</i>
WSMO	<i>Web Service Modeling Ontology</i>
WSRF	<i>Web Services Resource Framework</i>
WWW	World Wide Web
XML	<i>eXtensible Markup Language</i>

SUMÁRIO

1. INTRODUÇÃO.....	1
1.1. Contexto específico da pesquisa.....	7
1.2. Objetivo	9
1.3. Organização do texto	9
2. REVISÃO BIBLIOGRÁFICA.....	11
2.1. Arquitetura Orientada a Serviços.....	11
2.1.1. Serviços na Internet	13
2.1.2. Linguagem de Descrição dos Serviços na Internet.....	15
2.1.3. Composição Dinâmica de Serviços.....	18
2.1.4. SOA e Sistemas Produtivos Dispersos	19
2.2. Rede de Petri.....	29
2.2.1. Definição da Rede de Petri	31
2.2.2. Grafo de Fluxo da Produção	36
2.2.3. Procedimento PFS/RdP	37
2.3. Síntese do capítulo.....	38
3. ARQUITETURA DE CONTROLE PARA ORQUESTRAÇÃO DE SPs DISPERSOS.....	40
3.1. Fundamentos da arquitetura proposta	40
3.2. Descrição Geral da arquitetura proposta	42
3.2.1. Integrador de SPs	44
3.2.2. Serviço de Integração e Descoberta	44
3.2.3. Cliente / Interface Cliente	47
3.2.4. Serviço de Coordenação	47
3.3. Diagramas de Casos de Usos	48
3.4. Síntese do Capítulo	49
4. ESPECIFICAÇÃO DOS COMPONENTES/SERVIÇOS DA ARQUITETURA PROPOSTA.....	50
4.1. Serviço de Integração e Descoberta.....	50
4.1.1. Atividades de [Cadastro de SPs] e [Funções de Cadastro]	50
4.1.2. Atividades [Atualiza Estados] e [Funções de Atualização de Estados].....	54
4.1.3. Atividades [Solicitação de SPs] e [Funções de Solicitação de SPs]	58
4.1.4. Atividades [Solicita Único SP] e [Funções de Solicitação de único SP]	62
4.2. Serviço de Coordenação	66

4.2.1.	Atividades [<i>Novo Pedido</i>], [<i>Seleciona Critério</i>] e atividades [<i>Funções de Registro de Pedido</i>] e [<i>Funções de Registro de Critério para Pedido</i>]	66
4.2.2.	Atividade [<i>Consulta Pedidos</i>]	73
4.2.3.	Atividade [<i>Disponibilidade de SPs</i>]	75
4.2.4.	Atividade [<i>Execução Etapa SPx</i>]	78
4.2.5.	Atividade [<i>Resposta Serviço SPx</i>]	81
4.2.6.	Atividade [<i>Registra Histórico Pedido</i>]	83
4.3.	Síntese do capítulo	85
5.	EXEMPLO	86
5.1.	SP Disperso considerado	86
5.2.	Serviço de Integração e Descoberta	88
5.3.	Serviço de Coordenação	91
5.4.	Discussão dos resultados	92
6.	CONCLUSÕES	94
	TRABALHOS FUTUROS	96
7.	REFERÊNCIAS	97

1. INTRODUÇÃO

Os *sistemas produtivos (SPs)* são definidos como sendo aqueles que realizam um processo utilizando recursos materiais, equipamentos, recursos humanos e outras entidades físicas, para a produção de bens ou serviços (VILLANI, MIYAGI e VALLETE, 2007) (GARCIA MELO, 2011) (PESSOA *et al.*, 2012). Neste sentido, os sistemas de manufatura integram o grupo de SPs que envolvem serviços e processos de transformação física de matéria-prima para agregar valor e gerar um produto.

Existem alguns pontos de vistas e classificações diferentes para a evolução dos SPs. De acordo com a abordagem proposta por (MEHRABI, ULSOY e KOREN, 2000) a evolução desses sistemas pode ser dividida em três períodos distintos: o pré-CNC; o CNC e o mais recente, a “época do conhecimento”. O período “Pré-CNC” se refere aos SPs existentes antes da década de 1960, em que a maioria das máquinas, em especial na área de manufatura, e seu controle eram manuais. Na produção, explorava-se o conceito de partes intercambiáveis e utilizavam-se linhas com transportadores para redução de custos. A competição era basicamente local, com pouca integração entre diferentes sistemas e também não havia demanda por variedade nos produtos, ou seja, era possível ter um longo e sustentável período com apenas um produto.

A “época CNC” (de 1960 a 1990) é marcada com a disseminação do controle automático e com o surgimento das máquinas com comando numérico (NC) e sua posterior evolução para as máquinas CNC (comando numérico computadorizado). Esta invenção teve impacto direto nas taxas de produção, aumentou a qualidade e precisão dos processos de fabricação, com um controle mais preciso de posicionamento e movimentação de ferramentas, além de mais fácil integração lógica e física entre processos, máquinas e sistemas. Consequentemente, o número de técnicas produtivas como os sistemas flexíveis de manufatura e as técnicas japonesas de produção como *Kaizen* (melhoria contínua), *Just-in-time* (redução de estoque para redução de custos), *lean manufacturing*, entre outras, atraíram considerável atenção (MEHRABI, ULSOY e KOREN, 2000) (RAHANI e AL-ASHRAF, 2012) (HERRON e HICKS, 2008). No entanto, como apontado por (GARRO e

MARTIN, 1993), para se ter rápida resposta às flutuações do mercado, era necessário que os SPs possuísssem uma estrutura modular com ferramentas que permitissem uma rápida reorganização de suas tarefas. Todavia, a infraestrutura existente nessa época, como *hardware*, *software*, controladores, comunicadores e as próprias máquinas, não permitiam que essas mudanças ocorressem de fato.

O terceiro período, a partir de 1990, definido por (MEHRABI, ULSOY e KOREN, 2000) como “época do conhecimento”, é caracterizado pela intensificação da competição global e o progresso nos computadores e tecnologias de informação. Este rápido progresso foi observado em áreas como o gerenciamento dos sistemas de informação, no desenvolvimento de *softwares* para propósitos diversos e implementação de novos processos de fabricação, nos avanços significativos nos sistemas de comunicação (*hardware* e *software*) e novos materiais, e a penetração da computação e da mecatrônica em vários campos. Dessa maneira, a competição global, a mecatrônica e a tecnologia da informação são as forças motoras por trás das recentes mudanças nos SPs. Cada vez mais os consumidores demandam por produtos personalizados, com alta qualidade a baixos custos, e com o menor *time-to-market* possível. Entretanto, apesar dos grandes progressos tecnológicos em relação ao período anterior, os SPs ainda não conseguem atender totalmente a essa demanda. Os principais problemas ainda são (CÂNDIDO *et al.*, 2009b):

- Longo tempo para o projeto e instalação dos SPs;
- Ainda difícil reconfiguração de SPs devido a variações de produtos e complexidade desta tarefa, isto é, há a necessidade de pessoas especializadas, que conheçam a parte mecânica, elétrica, de controle e de automação que compõem os SPs, e o tempo despendido para esta reconfiguração ser alto;
- Interação entre as implementações de controle de SPs que costumam ser extremamente centralizadas e hierarquizadas;
- Escalabilidade do sistema como um todo que devido ao número de componentes e a necessária interação entre eles envolve complexidade exponencial;

- Dificuldade de implementação do conceito de tolerância à falhas sem incremento de redundâncias;
- Incompatibilidade física, lógica e dos modelos de negócios das diferentes entidades envolvidas: fornecedores e prestadores de serviços;
- Falta de um padrão para comunicação e integração de dispositivos e SPs amplamente aceito;

Outra necessidade evidente para os SPs atenderem a nova demanda é a de que sejam altamente flexíveis, isto é, para que se consiga adaptá-los facilmente para atenderem uma maior variação de produtos; e que sejam enxutos, para que produzam sem desperdícios, evitando custos desnecessários. Neste contexto, “agilidade” para um SP implica em sua capacidade de ser mais que flexível e enxuto. Agilidade é um requisito fundamental para que os modernos SPs sejam capazes de enfrentar os desafios provocados pela globalização, restrições ambientais e de trabalho, melhorar padrões de qualidade e se adaptar as mudanças tecnológicas (CHING-TORNG LIN e PO-YOUNG, 2006). Contudo, para uma empresa global, a agilidade é sempre limitada por sua camada menos ágil, isto é, todas as camadas de um SP precisam ser ágeis e interagir de uma maneira eficiente e sincronizada (CÂNDIDO *et al.*, 2009b). Os SPs são frequentemente organizados em uma estrutura hierárquica, como mostra a Figura 1.1, cujas camadas funcionais são definidas pela norma IEC 62264 (GROBA *et al.*, 2008), sendo que, atualmente, as camadas 1 e 2 são as menos ágeis devido as limitações dos dispositivos disponíveis, em comparação com as demais camadas (CÂNDIDO *et al.*, 2009a).

Assim, trabalhos como os citados em (LEITÃO, 2009) indicam que para se adaptar a este tipo de mercado, as empresas não devem atuar independentemente, isto é, devem reconsiderar a maneira como se organizam para aumentar sua competitividade. Para evoluir a ambientes mais ágeis, as empresas têm se focado em suas competências principais e assim precisam interagir com outras para prover especialidades complementares e recursos, o que resulta na distribuição de funções. O crescimento da complexidade dos novos produtos também contribui com a descentralização da produção, isto é, a dispersão física dos SPs, por fazer com que as empresas precisem colaborar, umas com as outras, na produção, na cadeia de suprimentos ou em redes de serviços, em que cada parceiro na cadeia ou na rede é

efetivamente responsável pela produção do produto final (GREFEN *et al.*, 2009). Consequentemente, a pesquisa em controle dos SPs tem mudado sua abordagem tradicional centralizada para uma estrutura em que os SPs ou parte desses SPs ficam espalhados por todo o mundo (JUNQUEIRA, VILLANI e MIYAGI, 2005), tornando-se um sistema disperso com distribuição não só das instalações físicas como também das funções. Neste texto o termo “sistema disperso” é adotado já que o termo “sistema distribuído” é comumente usado apenas para um tipo de sistema com funções distribuídas em subsistemas independentes, mas, eventualmente instaladas próximas fisicamente (TANENBAUM e STEEN, 2007).

Figura 1.1 - Estrutura típica de camadas de SPs. Adaptado de (GROBA *et al.*, 2008)

Neste contexto, em um ritmo crescente e de grande popularização, a tecnologia da Internet emerge como a base para interconectar dispositivos nos mais vastos e diversos domínios de aplicações (CÂNDIDO *et al.*, 2009a). Esta tendência é resultado de algumas evoluções convergentes como, por exemplo, (JAMMES e SMIT, 2005):

- Disponibilidade da tecnologia de Internet com baixo custo e alto desempenho;
- Aceitação do protocolo de comunicação TCP/IP entre computadores, como meio padrão para troca de informações entre dispositivos conectados.

Adicionalmente, a SOA (*Service-Oriented Architecture*) é uma abordagem que traz novas perspectivas para o atual ambiente de SPs (CÂNDIDO *et al.*, 2009b). SOA estabelece um conjunto de princípios de uma arquitetura de sistema autônomo e interoperável, que tem por objetivo aumentar a eficiência, agilidade e produtividade de um sistema por meio da adoção generalizada do conceito de serviços. De um modo genérico, um serviço pode ser definido como a execução de um trabalho ou uma função de “um” para “outro”. Do ponto de vista computacional, os serviços encapsulam funcionalidades de uma maneira transparente, de modo que podem ser descritos, publicados, alocados e dinamicamente invocados independentemente do sistema e linguagem de programação utilizada (GROBA *et al.*, 2008). Dessa maneira, SOA apresenta-se como uma arquitetura modular que encapsula o funcionamento, ocultando a heterogeneidade das partes do sistema, permitindo, desta forma, características de flexibilidade, confiabilidade e fácil implementação de soluções de *middleware*¹ (GROBA *et al.*, 2008), como observado na Figura 1.2. Atualmente, os *Web Services* (WSs) são a tecnologia mais adotada para implementar uma aplicação computacional em SOA. Os WSs podem ser vistos como uma tecnologia padrão para facilitar a interoperabilidade, integração e reuso de componentes da aplicação, isto é, serviços. Ao implementar a SOA por WSs em todas as camadas de um SP, isto é, desde os níveis de negócio (camada 4 da Figura 1.1) aos dispositivos de campo (camada 1) tem-se expectativas promissoras para se superar os desafios enfrentados pelos SPs dispersos, tornando-os mais ágeis e reconfiguráveis às mudanças de mercado, além de proporcionar uma maneira de integração e colaboração entre os SPs (que compõem o SP disperso).

Entretanto, para a implementação de um SP disperso sob a abordagem de SOA é fundamental assegurar uma interação eficaz de todas as partes envolvidas, isto é, uma “orquestração” dos processos produtivos vistos como serviços. **A**

---

¹ *Middleware* é uma aplicação computacional com ferramentas e interfaces disponíveis para comunicar-se com outras aplicações em outras camadas.

orquestração é entendida como uma abordagem centralizada para controle de um grupo de lógicas de processo que permite a interoperabilidade entre duas ou mais aplicações. A orquestração consiste em algumas regras, condições e eventos que especifica como diferentes parceiros devem interagir com um nó central para realização de uma tarefa (CÂNDIDO *et al.*, 2009b). Esta orquestração, dentro da SOA, garante assim a flexibilidade estrutural considerando as características heterogêneas dos componentes (SPs) que integram o SP disperso.

Figura 1.2 - Exemplo de um *middleware* integrando camadas de um SP. Adaptado de (GROBA *et al.*, 2008)

Existem assim algumas propostas de implementação de arquiteturas de sistemas de controle considerando um ambiente distribuído e disperso de sistemas produtivos (SPs) (GARCIA MELO, 2011), (PESSOA *et al.*, 2012) mas, em geral, o conjunto de SPs considerado é tratado como imutável durante a execução dos processos produtivos. Por outro lado, para assegurar a flexibilidade e agilidade de um SP, do ponto de vista prático, é necessário considerar a disponibilidade de SPs com funcionalidades semelhantes já que sempre existe a possibilidade de indisponibilidade (temporária ou não, parcial ou não, etc.) desses sistemas por algum motivo que não é domínio do orquestrador. Neste contexto, um orquestrador de um sistema que envolve SPs com funções semelhantes deve também considerar

políticas de prioridade para a seleção do SP na designação das tarefas a serem efetivamente realizadas.

Por fim, na definição e especificação da arquitetura do sistema de controle de SPs, o uso de uma abordagem de modelagem baseada em uma ferramenta formal, gráfica e executável, como é a rede de Petri (RdP) e suas derivações, permite estruturar um procedimento para a construção dos modelos e também para a análise funcional das partes que compõem este sistema e a interação entre elas (MELO *et al.*, 2010), bem como as propriedades do sistema modelado (ZURAWSKI e ZHOU, 1994), (CARDOSO; VALETTE, 1997). Baseado na natureza deste tipo de arquitetura de sistema de controle, pode-se classificá-lo como um sistema a eventos discretos (SED) e, assim, técnicas derivadas de Redes de Petri (RdP), incluindo a *Production Flow Schema* (PFS), podem ser utilizadas em uma abordagem PFS/RdP para modelagem (MIYAGI, 1996). Esta abordagem tem se mostrado eficiente para descrições hierárquicas, análise e controle de SEDs (HASEGAWA, 1996), (ZHOU e VENKATESH, 1999). O sistema é abordado em diferentes níveis de abstração: um modelo conceitual é obtido através da técnica PFS e um modelo funcional através da RdP e, aplicado a modelagem da arquitetura de controle, permite a caracterização tanto dos serviços que integram esta arquitetura como a interação entre eles.

1.1. CONTEXTO ESPECÍFICO DA PESQUISA

Baseado no que foi anteriormente apresentado, este trabalho foi desenvolvido com o contexto específico indicado a seguir.

Premissas assumidas:

1. SPs dispersos são encapsulados e expostos como um serviço. Isto é, existe uma aplicação que expõem as funcionalidades deste SP através de um serviço. Desta maneira, a orquestração de SPs proposta é realizada através destes serviços (Figura 1.3). No trabalho desenvolvido por (MELO *et al.*, 2010) foi proposta uma abordagem sistemática para se expor SPs como serviço.

Figura 1.3 – Ilustração de Orquestração de SPs expostos como serviço

2. Como a orquestração de SPs é realizada através dos serviços que expõem estes SPs, a comunicação é realizada entre aplicações computacionais e, por isso, o tempo de resposta de equipamentos e controladores dos SPs não influencia na execução da orquestração. Isto é, os tempos de latência da rede e de processamento são de grandezas muito menores que os tempos de usinagem e outros processos de fabricação mecânica, por exemplo.
3. Arquiteturas de controle de SPs sob o conceito de SOA consideram que a alocação de tarefas para os SPs é estática, o que implica que em caso de indisponibilidade, deve-se aguardar a disponibilidade deste SP para dar continuidade ao processo produtivo. Assume-se que, desta maneira, a possibilidade de substituição de um SP indisponível por outro SP de mesma funcionalidade, é fundamental para assegurar maior versatilidade e agilidade ao sistema de controle de SPs dispersos para a execução do processo produtivo.
4. De acordo com a definição já apresentada para SPs, i.e., sistemas que utilizam recursos para produção de bens e serviços, entende-se que o conceito de SP disperso permeia diferentes níveis de abstração: como uma fábrica completa, uma linha de produção, uma célula de manufatura, ou até mesmo um único equipamento. Num SP disperso, os diferentes SPs interagem em um processo produtivo global, a fim de se atender a um pedido.

A metodologia de pesquisa adotada no presente trabalho é a metodologia de desenvolvimento de pesquisas, citada em (JENSEN, 1992). Esta abordagem evidencia a forte relação entre teoria, ferramentas e aplicações, estando os três aspectos integrados no processo de desenvolvimento de uma pesquisa tecnocientífica e contribuindo sinergicamente nos processos associados a cada um deles. Uma caracterização tripartida, como a proposta em (JENSEN, 1992), é também seguida nas metodologias de projeto de sistemas. Citando outra abordagem, tem-se, por exemplo, o trabalho de (CALVEZ, 1993), que identifica três aspectos básicos para a resolução de um problema: os métodos, as ferramentas e as técnicas. As técnicas fornecem o suporte direto à implementação, enquanto as ferramentas são utilizadas para implementá-las e os métodos permitem evoluir da definição do problema até a sua implementação.

1.2. OBJETIVO

Este trabalho objetiva desenvolver uma arquitetura para o sistema de controle de sistemas produtivos (SPs) dispersos sob o conceito de orientação a serviços e com uso da abordagem formal da Rede de Petri/ *Production Flow Schema* para sua especificação. Mais especificamente, visa-se a concepção da solução de “orquestração” de SPs considerando casos de funcionalidades semelhantes, isto é, seleção de SP em substituição a um outro SP em caso de indisponibilidade deste de acordo com um critério prioritário e, outros fatores para assegurar a produção de produtos de acordo com as demandas especificadas e provendo com isto maior agilidade e versatilidade ao SP disperso.

1.3. ORGANIZAÇÃO DO TEXTO

O capítulo 2 apresenta a revisão bibliográfica onde são abordados os conceitos de arquitetura orientada a serviços (SOA) bem como *Web Services* (WS) e suas especificações como o WSDL (*Web Service Description Language*, especificação para descrição de WS), redes de Petri (RdP) e a metodologia

PFS/RdP para a modelagem, análise estrutural e comportamental dos componentes da arquitetura e, uma discussão de trabalhos propostos para o uso da SOA aplicada em SPs.

No capítulo 3 é apresentada uma proposta para a arquitetura de controle para orquestração de SPs dispersos. Os componentes e atores presentes na arquitetura de controle para orquestração, bem como a interação entre eles são descritos detalhadamente neste capítulo.

No capítulo 4 apresenta-se a aplicação do procedimento PFS/RdP para desenvolvimento dos modelos dos processos na arquitetura proposta no capítulo anterior. Os modelos gerados, depois de devidamente verificados e validados, são entendidos como a especificação para a implementação prática dos componentes/elementos da arquitetura proposta.

No capítulo 5 é apresentado um exemplo de implementação de algumas partes da arquitetura proposta e, de como as funcionalidades são acionadas para assegurar o comportamento e desempenho especificados do SP disperso.

Por fim, o capítulo 6 apresenta as conclusões da pesquisa, isto é, uma discussão das principais contribuições e limitações do trabalho realizado, assim como os trabalhos futuros que possam ser derivados.

2. REVISÃO BIBLIOGRÁFICA

Este capítulo apresenta os conceitos fundamentais, técnicas e ferramentas relacionadas, principalmente, a sistemas distribuídos com orientação a serviços vistos como um sistema a eventos discretos (SED), de modo que a rede de Petri (RdP) pode ser utilizada para sua modelagem e análise.

2.1. ARQUITETURA ORIENTADA A SERVIÇOS

De acordo com (JAMMES e SMIT, 2005), a SOA (*Service Oriented Architecture*) é um conjunto de princípios arquiteturais para a construção de sistemas autônomos e interoperáveis. As principais características de sistemas autônomos são que eles são criados independentemente de outros sistemas; que são capazes de atuar independentemente do ambiente em que estão; e que são “auto-contidos”. Já a interoperabilidade é propiciada por uma interface clara e abstrata que expõe o serviço para o seu ambiente, escondendo a implementação deste serviço.

No âmbito computacional, a definição proposta por (OASIS, 2006) é de que a SOA é um paradigma para organizar e utilizar capacidades computacionais (como processamento, armazenamento, filtragem e outras) distribuídas que podem estar sob controle de diferentes domínios.

Num contexto maior, assume-se que entidades possuem capacidades para resolver ou dar suporte a uma solução para os problemas que eles encontram no curso de seus negócios. É natural imaginar as necessidades de uma pessoa sendo atendidas pelas capacidades oferecidas por outra pessoa, ou em computação, os requisitos de um agente computacional de certo proprietário sendo atendidos por outro agente computacional pertencente a outro proprietário. Não há necessariamente uma correlação de um para um entre necessidades e capacidades, ou seja, uma capacidade não foi feita necessariamente para atender uma necessidade, mesmo que seja usada para atender essa. A granularidade das

necessidades e capacidades varia da fundamental à complexa e, uma necessidade pode requerer a combinação de várias capacidades enquanto uma simples capacidade pode atender mais do que uma necessidade.

A descrição da SOA baseia-se em um conceito central: o serviço. De uma maneira genérica, um serviço pode ser definido como a execução de um trabalho ou uma função de “um” para “outro”, e deve ser capaz de executar um trabalho para o “outro”, especificar um trabalho oferecido para o “outro” e, oferecer a execução de um trabalho para o “outro” (OASIS, 2006). Com o conceito de serviço, que também pode ser entendido como a implementação de uma capacidade computacional, um sistema de informação pode ser melhorado para um novo nível e ser mais adequado a ambientes interoperáveis e heterogêneos (WANG e HU, 2009).

A SOA pode ser considerada também uma plataforma de integração universal alavancando funcionalidades de *software*. A lógica de integração que é orientada a processos é separada da interação do usuário, ou seja, a interface com o usuário é feita de uma forma independente da forma como é feito o controle de atividades. A conectividade é acoplada pelos adaptadores, ou seja, um serviço comunica-se com outro por meio de um adaptador. Um *message broker* (corretor de mensagens) fornece um mecanismo para validação, transformação e filtro de dados para a SOA (MAYERL, VOGEL e ABECK, 2005).

Com as características acima, a SOA adota um tipo de acoplamento considerado "fraco" entre seus componentes, ou seja, a comunicação entre serviços independe da forma como esses foram criados, mas de adaptadores (SHAN *et al.*, 2009) (MAYERL, VOGEL e ABECK, 2005) (ZHU e MELO, 2009) (WANG e HU, 2009) e usa coleções de serviços reusáveis que podem ser recombinaados para definir novos processos ou refinar um já existente. A ideia fundamental é similar à de modelos baseados em agentes: um serviço de baixo nível pode ser parte de alguns serviços de alto nível ou de processos de negócios colaborando com outros serviços (MEDINA *et al.*, 2009) (WANG e HU, 2009) (ZHU e MELO, 2009). Ambientes fracamente acoplados apresentam uma redução de dependências entre aplicações com interação de estilo de mensagem, de semântica e sintaxe, e de modo assíncrono. A SOA assume a abordagem de *middleware* existente, e assim a implementação e as tecnologias específicas do detalhamento do serviço são

ocultadas atrás da definição de interfaces abstratas, privilegiando a autonomia e o fraco acoplamento dos serviços a serem integrados (MAYERL, VOGEL e ABECK, 2005). Os serviços na SOA são auto-contidos, ou seja, executam tarefas pré-determinadas (ZHU e MELO, 2009) (WANG e HU, 2009). A SOA provê um padrão de desenvolvimento flexível independente da complexidade e da granularidade do serviço (SHAN *et al.*, 2009).

2.1.1. Serviços na Internet

A tecnologia de serviços na Internet, mais conhecida como *Web Service* (WS) é citada como meio mais utilizado para a implementação da SOA. O WS é um conjunto de aplicações (programas de computador) que encapsulam operações de modo a compor módulos. Por meio de WS, as aplicações podem ser descritas, publicadas, localizadas e invocadas em uma rede de comunicação tipo WWW (*World Wide Web*) (GOTTSCHALK, 2000) e, é possível considerar esta abordagem como uma instância da SOA. Um sistema que adota o WS envolve os componentes a seguir (Figura 2.1):

Figura 2.1 - Componentes do Web Service. Adaptado de (GOTTSCHALK, 2000)

- Fornecedor do serviço (*Service provider*): que é o provedor de serviço (de processamento de informação) e representa a aplicação que hospeda o WS permitindo que os “clientes” acessem o serviço e disponibilizem os serviços no repositório de descrição de serviços.

- Repositório de descrição de serviços (*Service register*): representa os servidores de registro e busca de WS baseados em arquivos de descrição de serviços que foram disponibilizados pelos fornecedores dos serviços, facilitando a localização deste, via internet, com os consumidores dos serviços.
- Consumidor do serviço (*Service requester*): que é o “cliente” do serviço e representa a aplicação que está procurando, invocando ou iniciando uma interação com o WS. O cliente do serviço pode ser uma pessoa acessando o sistema por meio de um navegador ou uma aplicação computacional realizando uma invocação aos métodos descritos na interface do WS e pelo qual busca informação no repositório de descrição de serviços.

Em uma visão alternativa, o WS envolve uma configuração hierárquica de protocolos de serviço (Figura 2.2), os quais estão agrupados em camadas de serviços (CERAMI, 2002), a saber:

- Serviço de Transporte:
Esta camada é responsável pelo transporte de mensagens entre aplicações computacionais. Atualmente, esta camada inclui o HTTP² (*Hyper Text Transfer Protocol*), SMTP³ (*Simple Mail Transfer Protocol*), FTP⁴ (*File Transfer Protocol*), e protocolos mais recentes, tais como o BEEP⁵ (*Blocks Extensible Exchange Protocol*).
- Serviço de Mensagem:
Esta camada é responsável pela codificação das mensagens em um formato XML⁶ (*eXtensible Markup Language*). Essa linguagem permite além de descrever uma informação, apresentar sua estrutura. Adicionalmente, o formato pode ser compreendido pelo homem. Atualmente, essa camada inclui o RPC⁷ (*Remote Procedure Call*) e o SOAP⁸ (*Simple Object Access Protocol*),

---

²<http://www.w3.org/Protocols/>

³<http://www.networksorcery.com/enp/protocol/smtp.htm>

⁴<http://www.networksorcery.com/enp/protocol/ftp.htm>

⁵<http://www.beepcore.org/>

⁶<http://www.w3.org/TR/REC-xml/>

⁷<http://penta.ufrgs.br/rc952/trab1/rpc.html>

- Serviço de Descrição:

Esta camada é responsável pela descrição de uma interface pública para um determinado serviço (do WS). Atualmente, o serviço de descrição pode ser realizado, por exemplo, por meio do WSDL⁹ (*Web Service Description Language*).

- Serviço de Aplicação:

Esta camada programa as funcionalidades específicas do serviço oferecido. A implementação desta camada pode ser feita em diferentes linguagens, tais como: Java, C#, Visual Basic, entre outros.

- Serviço de Coordenação:

Esta camada é responsável pelo fluxo dos processos. O fluxo dos processos define o número de etapas que devem ser executadas e o conjunto de condições que determinam a ordem natural destas etapas. Neste sentido, esta camada envolve os programas de gerenciamento para permitir a integração e harmonização das atividades, em cada etapa, dos diferentes serviços envolvidos no desenvolvimento do fluxo dos processos. Atualmente, o serviço de coordenação tem sido abordado por meio do BPEL (*Business Protocol Executable Language*).

Serviço de registro - UDDI	Serviço de Coordenação - BPEL
	Serviço de Aplicação - Java, C#,...
	Serviço de Descrição - WSDL
	Serviço de Mensagem - SOAP
	Serviço de Transporte - HTTP

Figura 2.2 - Protocolos de serviços para a implementação de WS (GARCIA MELO, 2011).

2.1.2. Linguagem de Descrição dos Serviços na Internet

Como os protocolos de comunicação e os formatos das mensagens são padronizados, torna-se cada vez mais necessário e importante ser capaz de

---

⁸<http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

⁹<http://www.w3.org/TR/wsdl>

descrever estas comunicações de uma maneira estruturada (CHRISTENSEN *et al.*, 2001). Neste contexto, surge o Web Service Description Language (WSDL) como um documento em XML que realiza a descrição de WS e como este WS é acessado, isto é, ele descreve a localização do WS (endpoints) e as operações (ou métodos) que este serviço expõe¹⁰. As definições de serviço contidas no WSDL provêm a documentação e servem como receita para automação de funções envolvendo a comunicação de aplicações (CHRISTENSEN *et al.*, 2001).

O WSDL define os seguintes elementos na descrição de WS:

- **Types (tipos)** – um componente que contém as definições dos tipos de dados usados pelo WS, isto é, contém todas as informações necessárias para a troca de informação entre o consumidor de serviço e o fornecedor do serviço. Permite ainda o uso de esquemas em XML externos (como por exemplo, o *XSD – XML Schema Definition*);
- **Message (mensagem)** – uma definição de tipo abstrato para os dados que estão sendo comunicados;
- **Operation (operação)** – uma descrição abstrata da ação suportada pelo serviço. Este elemento define as ações em SOAP e a codificação da mensagem, bem como os parâmetros de entrada e saída para a operação;
- **Port Type/ Interface (tipo de porta)** – define um conjunto de operações abstratas que suportam um ou mais *endpoints*. Este elemento define o WS e cada operação que a porta expõe;
- **Binding (conexão)** – define o protocolo concreto e uma especificação para o formato de dados de um *port type* específico. Este elemento define o estilo da conexão em SOAP e o protocolo de transporte para o WS (*port type* ou *interface*);
- **Port (porta)** – especifica um único *endpoint* definido como uma combinação de conexão com um endereço de rede de comunicação.

---

¹⁰ http://www.w3schools.com/wSDL/wSDL_intro.asp

Ele define o ponto de conexão com o WS e, é usualmente representado por uma *string* de uma HTTP URL;

- **Service (serviço)** – uma coleção de *endpoints* relacionados;

O WSDL também define um mecanismo de conexão (*binding*) comum para todos os WSs, que é utilizado para adicionar um protocolo específico para o formato de dados, ou uma estrutura para uma mensagem abstrata, operação ou *endpoint*. O WSDL permite o reuso de definições abstratas. O WSDL também utiliza *binding extensions* (extensões de conexão) específicas para os seguintes formatos de mensagem e protocolos: SOAP 1.1, HTTP GET/POST, MIME.

A recomendação atualmente feita pelo W3C (MINOR, 2012) é utilizar o WSDL 2.0, mas o WSDL 1.1 continua sendo vastamente empregado. As estruturas dessas WSDL são relativamente similares, como ilustra a Figura 2.3.

Figura 2.3 - Estrutura dos documentos WSDL 1.0 e WSDL 2.0 (MINOR, 2012)

Um arquivo WSDL é uma forma de contrato de serviço. Através do WSDL, o WS fornece sua interface exterior detalhada, consistente com os princípios do SOA.

Ao localizar e analisar um arquivo WSDL, um cliente tem todas as informações necessárias para usar as operações fornecidas pelo WS (MINOR, 2012).

Existem outras linguagens para a descrição de serviços como a DAML-S (*DARPA agent mark-up language for services*) e, posteriormente, a OWL-S (*Ontology Web Language for services*), a WSMO (*Web Service Modeling Ontology*), WSDL-S (WSDL com semântica), a USDL (*Unified Service Description Language*), etc. Uma revisão das linguagens de descrição de serviços pode ser encontrada em (SUN, DONG e ASHRAF, 2012). Ainda assim, o padrão mais amplamente aceito é o WSDL padronizado pela W3C (SUN, DONG e ASHRAF, 2012).

2.1.3. Composição Dinâmica de Serviços

A estrutura mais frequente de invocação de serviços na internet é construída de forma estática, sempre referenciando o mesmo tipo de WS e os mesmos métodos/operações destes WSs (PINTO *et al.*, 2008). Também, no contexto de SPs dispersos, muitos trabalhos consideram que os tipos de SPs envolvidos em determinado processo produtivo são sempre os mesmos, isto é, estáticos (PESSOA *et al.*, 2012), (GARCIA MELO *et al.*, 2010) (GARCIA MELO, 2011), etc. A orquestração dos serviços dos SPs visa a execução do processo produtivo global e, assim, atender ao pedido do cliente, no entanto, durante a execução dos processos, para que seja possível a seleção de SP em substituição a um outro SP, de mesma funcionalidade, é necessário considerar conceito de composição dinâmica de serviços.

A composição dinâmica de serviços é um tema complexo e existem diversos trabalhos que o abordam. Em (D'MELLO, ANANTHANARAYANA e SALIAN, 2011) é apresentada uma revisão das diferentes técnicas para a composição dinâmica de WS. Outros trabalhos tratam a composição dinâmica de WSs por meio de sua ontologia e realizam a descoberta e identificação semântica destes WSs, (ZHANG, ARPINAR e ALEMAN-MEZA, 2003), (TRAVERSO e PISTORE, 2004), (DE SOUZA e RABELO, 2011), entre outros. Face ao escopo e objetivo do presente trabalho, em que cada entidade, isto é, o serviço do SP disperso, precisa de um certo grau de

autonomia, se adequa a abordagem de composição de WS realizada apenas pela avaliação sintática dos WSs, por meio de seus WSDLs.

Neste contexto, em (MARTINS, 2007) (PINTO *et al.*, 2008) foi apresentada uma arquitetura de WS com suporte a composição dinâmica de serviços, baseada em padrões de projeto (Figura 2.4). Nesta arquitetura, um cliente de um serviço não realiza a chamada diretamente ao WS desejado, e sim a um “agente” que redireciona a chamada ao WS com maior disponibilidade. O “diretório ativo de serviços” contém a lista de serviços publicados e notifica a “aplicação”, que é quem consome o serviço, toda vez que um novo serviço é publicado. O registro de um serviço no “diretório ativo de serviços” é feito por meio da chamada de um método em que é passada sua localização (URL) e qual o tipo de serviço esperado. O cliente, por sua vez, é visto como composto de duas partes: a “aplicação”, que consome o serviço propriamente dito e; o “agente”, que realiza a ligação entre a aplicação e o serviço escolhido.

Figura 2.4 – Arquitetura de um WS com suporte a composição dinâmica (MARTINS, 2007)

2.1.4. SOA e Sistemas Produtivos Dispersos

Apesar da SOA ser concebida com o foco em ambientes puramente computacionais, seu potencial de aplicação em outras áreas é evidente. Assim, na última década, foram realizados trabalhos com o objetivo de avaliar a utilização da

SOA em aplicações para a automação industrial e colaboração de SPs. Surgiram algumas iniciativas de pesquisa visando aplicar a SOA no âmbito industrial. Entre as principais iniciativas que foram identificadas nas bases de dados do *Web of Science* e *Scopus* tem-se: IMC-AESOP Project (IMC-AESOP Project, 2011) IST SOCRADES (SOCRADES, 2008), ITEA SODA (SODA, 2007), OPC-UA (OPC, 2008), STREP InLife (INLIFE, 2007), RI-MACS (RI-MACS, 2007), STREP CoBIs (COBIS, 2006) e SIRENA (SIRENA, 2006). Em (GARCIA MELO, 2011) e (CÂNDIDO *et al.*, 2009b) pode-se encontrar uma revisão detalhada destes trabalhos. A seguir é apresentada uma perspectiva geral sobre os principais trabalhos publicados.

Uma abordagem visando a integração entre fornecedores, consumidores, clientes e concessionários de um sistema industrial automotivo é apresentada em (HOLMQVIST, DURCHSLAG e BROWN, 2004). Neste trabalho, os autores desenvolvem um sistema de informação baseado na integração de processos encapsulados em WSs e, que permite ativar o processo produtivo de fabricação das peças de automotores como resposta à demanda das concessionárias e clientes da indústria automotiva.

Em (YAN, LIANG e DU, 2005) é apresentado uma arquitetura para implementar o controle remoto de instrumentos usando WSs com o intuito de disponibilizar laboratórios *on-line*. A arquitetura apresentada é composta de três camadas: (1) uma camada superior relacionada com os conceitos de negócio, onde são estabelecidas as especificações do processo e do gerenciamento do laboratório; (2) uma camada intermediária que trata serviços específicos por meio de uma rede de comunicação local (LAN), ou uma externa, utilizando internet; (3) uma camada inferior de aplicação do serviço, a qual trata do processamento de dados do laboratório remoto.

Uma arquitetura para a execução de serviços também é apresentado em (MADHUSUDAN, 2005). Seu trabalho centra na seleção, execução e monitoração de um WS. O WS neste caso é composto por outros WSs. A execução das atividades dos WSs segue a definição de um processo descrito em BPEL, o qual é selecionado de um repositório baseado na requisição de um cliente e disponibilidade do WS que o compõe.

Em (KUMAR *et al.*, 2005) o processo de integração das funcionalidades de um SP é chamado de “virtualização do SP”. Para disponibilizar as funcionalidades do SP como um serviço, este trabalho utiliza um conjunto adicional de padrões para a definição do WS, como o WSRF¹¹ (*Web Services Resource Framework*), o qual permite a monitoração do estado do serviço. Para a virtualização de um componente são propostos quatro pontos de vista do dispositivo: físico, operacional, cumprimento e, econômico. O físico considera aspectos técnicos de configuração, o operacional considera aspectos funcionais, o de cumprimento aglomera mecanismos para solução de conflitos para a realização de uma tarefa e, o econômico considera o custo de utilização desse recurso. Neste trabalho, é apresentado um exemplo de aplicação da disponibilização dos serviços de um AGV (*Automatic Guided Vehicle*) que foi implementado em linguagem Java. Também adotando o conceito de virtualização, em (SUA, GUOA e CHANGB, 2005) é apresentada uma SOA com o objetivo de permitir a reconfiguração de um serviço chamado DMSPM (*Dynamic Manufacturing Service Provision Mechanism*). A infra-estrutura apresenta uma estrutura hierárquica de três camadas: (1) camada superior, que disponibiliza as funcionalidades do sistema de manufatura ao usuário e, desta forma, recebe os pedidos do usuário; (2) camada intermediária, de coordenação de serviços, que reúne os diferentes processos de negócios, tais como os de produção, logística, manutenção, entre outros e; (3) camada inferior, onde são disponibilizadas as funcionalidades dos recursos do SP como um serviço. Em (MENG e YEXIN, 2005) é apresentada a arquitetura SOMG (*Service Oriented Manufacturing Grid*) baseada em um padrão chamado GRID¹², que permite a interação de um conjunto de serviços (dos WS) estruturados na forma de uma rede com a conservação do estado. A arquitetura apresentada adota uma estrutura multicamada: (1) camada inferior, chamada “recursos de manufatura”, em que são disponibilizados os serviços básicos dos recursos do sistema, tais como: equipamentos, espaços de armazenagem, entre outros; (2) camada intermediária, chamada de “agregação”, onde são disponibilizados os meios para o registro, procura e descoberta dos serviços disponibilizados na camada 1, permitindo uma coordenação dos serviços; (3) camada de execução onde uma rede dos serviços do sistema são disponibilizados

---

¹¹http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf

¹²<http://www.globus.org/toolkit/>

baseados nos requerimentos de negócio; (4) camada de supervisor que permite a configuração do sistema e; (5) camada de apresentação onde são disponibilizadas as funcionalidades do sistema aos usuários. Em uma abordagem similar para sistemas de manufatura, (LIU *et al.*, 2006) apresentam uma estrutura para coordenar o uso compartilhado dos recursos, considerando a heterogeneidade e autonomia relativa de cada um dos componentes, além da organização dos serviços de manufatura fornecidos. Nesta abordagem, uma camada de recursos disponibiliza os serviços de cada componente e uma camada de abstração coordena os serviços dos componentes.

Em (KIM *et al.*, 2006) é apresentada uma SOA, chamada *WSCPC (Web Services for Collaborative Product Commerce)*, que permite a colaboração entre usuários na seleção do sistema de manufatura necessário para atender suas necessidades. Neste sentido, *WSCPC* procura o fornecedor dos serviços necessários em seu repositório, agenda a atuação de cada serviço para atender ao pedido dos clientes e, coordena essas atividades. A abordagem selecionada para a coordenação das atividades é a de coreografia¹³, implementada com o padrão *WSCI (Web Service Choreography Interface)*. Esta arquitetura considera quatro componentes: (1) uma interface de portal de usuário que permite formatar a requisições dos clientes; (2) um servidor de processo de negócios que desenha o processo de negócios baseado nas requisições do usuário; (3) um módulo de informação para atender os serviços de informação dos pedidos e; (4) um módulo de serviços que disponibiliza mecanismos para o registro e descoberta de serviços de sistemas de manufatura e ligação aos serviços registrados. Em (LI-FENG e ZHOU, 2007) é apresentada uma arquitetura para o controle de um sistema flexível de manufatura. Neste trabalho é utilizado o padrão de comunicação *MMS*¹⁴ (*Manufacturing Message Specification*) e *WSs* para disponibilizar os serviços dos equipamentos. *MMS* é uma especificação tanto da descrição dos dispositivos físicos como do intercâmbio de informações. Adicionalmente, esta especificação define o formato das mensagens. Assim, a arquitetura considera dois níveis, um de *WSs* que

---

¹³ Coreografia é a interação entre componentes distribuídos sem a existência de uma entidade controlando a lógica de colaboração.

¹⁴<http://www.sisconet.com/downloads/mmsovr1g.pdf>

apresentam as funcionalidades dos equipamentos como serviços e que enviam mensagens segundo as especificações MMS. O outro nível, de aplicações computacionais de usuários, que coordena os serviços dos equipamentos de acordo como pedido dos clientes. Em (KOMODA, 2006) são referenciadas as SOAs para o gerenciamento de um comboio de carros e o controle de uma máquina de fabricação de semicondutores. A estrutura do sistema de gerenciamento do comboio de carros é hierárquica onde, em um nível inferior, são disponibilizados os serviços básicos fornecidos pelos dispositivos ou equipamentos tais como carros registrados e mecanismos para o registro e descoberta de novos dispositivos e equipamentos; em um nível intermediário, tem-se o serviço de coordenação que permite a integração dos serviços do nível inferior, o qual garante a consistência dos dados trocados e; em um nível superior, encontram-se aglomerados os serviços independentes, tais como: de gestão, controle de operações e gerenciamento dos dados, entre outros. Por sua vez, a arquitetura de controle do equipamento para a fabricação de semicondutores precisa ser flexível para dar resposta às mudanças tecnológicas dos semicondutores. Considerando que a estrutura física do equipamento dispõe de uma área de transferência de partes na qual atuam um número variável de braços mecânicos, o sistema de controle da máquina é integrado por um conjunto de módulos de controle, um para o gerenciamento das atividades da área de transferência chamado TM (*Transfer Module*), e outros que controlam as atividades de cada braço, chamados de PMCs (*Process Module Controller*). O TM é um serviço que apresenta interfaces para os comandos do operador e para a conexão com os PMCs de forma dinâmica. Desta forma, este nível supervisiona os movimentos dos braços mecânicos. Em (DELAMER e LASTRA, 2006) é apresentado um trabalho relacionado com a orquestração de serviços, o qual coordena as atividades de vários equipamentos em um sistema de manufatura onde as funções são encapsuladas como WSs. Os autores se baseiam nas semânticas definidas dos WSs no contexto de uma ontologia para a composição de serviços e na configuração de um serviço de manufatura no nível de equipamentos. O termo ontologia, neste caso, referencia uma forma de organizar um conjunto de conceitos dentro de um domínio e os relacionamentos entre estes.

(BINDIETRICH, KIRN e SUGUMARAN, 2007) apresentam um estudo de caso da geração do processo de negócios considerando a orientação de serviços dos

fornecedores na produção de sapatos, o qual é implementado utilizando BPEL. Em (ZEEB *et al.*, 2007) é apresentado um trabalho de implementação da disponibilidade dos serviços de dispositivos embarcados utilizando o padrão DPWS¹⁵ (*Devices Profile for Web Services*). O DPWS estabelece um mínimo de restrições para garantir a prestação de um serviço por parte de dispositivos utilizando um conjunto de especificações que garantem a execução dos serviços e envolve transmissão de mensagens com protocolo UDP¹⁶ (*User Datagram Protocol*) via internet. A implementação é baseada em outra implementação chamada gSOAP (ferramenta para se desenvolver WS baseado em SOAP em C/C++) sendo ambas realizadas em linguagem C++. Outras abordagens com o padrão DPWS são referenciadas em (KIRKHAN *et al.*, 2008) com o foco na disponibilização das funcionalidades de dispositivos físicos mediante uma abordagem SOA. Inicialmente, o autor referencia um projeto apoiado pela União Européia chamada SIRENA (SIRENA, 2006), com o objetivo de definir uma infraestrutura que permite disponibilizar as funcionalidades de uma rede de dispositivos como um serviço. Os resultados deste projeto têm sido base dos projetos SODA (SODA, 2007) , SOCRADES (SOCRADES, 2008) e IMC-AESOP (IMC-AESOP Project, 2011) também apoiados pela União Européia.

O objetivo do projeto SODA é desenvolver uma infraestrutura que permita uma intensa interação dinâmica entre dispositivos. Assim, a arquitetura definida utiliza o padrão DPWS e permite tanto a definição das funcionalidades do dispositivo como um serviço como a agregação das funcionalidades dos dispositivos para realizar atividades mais complexas. Adicionalmente, é proposto um padrão para a definição dos serviços dos dispositivos para facilitar sua descoberta.

O projeto SOCRADES, aborda particularidades dos dispositivos no ambiente industrial. Assim, a arquitetura proposta é hierárquica de quatro camadas: (1) camada inferior que disponibiliza as funcionalidades dos dispositivos na rede de comunicação como serviços; (2) camada acima que permite a agregação de dispositivos e definição de novas funcionalidades como máquinas; (3) camada com os serviços de processos, que define a sequência de execução dos serviços envolvidos e; (4) camada superior, em que uma aglomeração de serviços é

---

¹⁵<http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01>

¹⁶<http://www.faqs.org/rfcs/rfc768.html>

instanciada pelos usuários. Em (MOREIRA *et al.*, 2008) é apresentada uma arquitetura hierárquica no contexto do projeto SOCRADES. A arquitetura é composta de quatro camadas: (i) equipamentos; (ii) integração; (iii) aplicações comerciais de gerenciamento integrado de manufatura da SAP¹⁷ e; (iv) aplicações *ad hoc*. A camada de equipamentos envolve a disponibilização de serviços nos dispositivos no chão de fábrica de acordo com o DPWS. A camada de integração fornece serviços adicionais tais como geração de históricos, descoberta de serviços, orquestração dos serviços, monitoração de serviços e, dispõe de interfaces que viabilizam a interligação com aplicações integradas de manufatura fornecida pela SAP ou aplicações *ad hoc*.

Em (CIANCETTA, D'APICE e GALLO, 2007) uma SOA é definida para disponibilizar dinamicamente os serviços de dispositivos físicos. Nesse trabalho é utilizado o protocolo DHCP¹⁸ (*Dynamic Host Configuration Protocol*) para uma configuração dinâmica dos IPs (*Internet Protocol*) de cada sensor no momento em que cada um se conecta à rede de comunicação. A arquitetura proposta é composta de quatro camadas: CL (*Central Lab*); PLs (*Peripheral Lab*); FL (*Field Labs*); FI (*Field Instrument*). A camada CL coordena as funcionalidades dos laboratórios dos dispositivos na camada PL e, conforme seja necessário realiza as requisições das funcionalidades dos laboratórios na camada FL, ou instrumentos de campo aglomerados na camada FI. Em (MENDES *et al.*, 2007) é apresentada a arquitetura AWS (*Agent-based Web Services*) para o desenvolvimento de um sistema coordenado de manufatura. As funcionalidades de cada componente ou dispositivo do sistema são agrupadas em três níveis de serviços: (1) gerenciamento, (2) operação e, (3) eventos, os quais interagem entre si para atingir o objetivo do sistema. Os serviços no nível (1) permitem a configuração dos componentes, os serviços no nível (2) apresentam as funcionalidades operacionais dos componentes e, os serviços no nível (3) supervisionam a ocorrência de eventos de alerta. A coordenação é realizada por uma aplicação computacional independente.

Um sistema de controle de um SP baseado na SOA também foi definido em (MENDES *et al.*, 2008). A arquitetura define três componentes principais: (1)

---

¹⁷<http://www.sap.com>

¹⁸<http://www.networksorcery.com/enp/protocol/dhcp.htm>

componente que aglomera os sistemas mecatrônicos, chamado MeC (*Mechatronic Components*); (2) componente que considera os processos, chamado PCC (*Process Control Components*); (3) componente que fornece ferramentas de suporte, chamado ISC (*Intelligence Support Components*). O MeC considera a combinação de recursos mecânicos, eletro-eletrônicos e computacionais que permitem o controle local de um dispositivo. Este componente apresenta as funcionalidades por meio de WSs. O PCC define a sequência de execução dos serviços disponibilizados nos componentes do MeC com base no processo que foi verificado por meio da modelagem em rede de Petri de alto nível. O ISC permite a solução de conflitos na execução dos processos requisitados pelo PCC. Em casos simples, PCC e ISC podem ser integrados.

Em (DOMNIC e KARNOUSKOS, 2008) é utilizada uma arquitetura com o intuito de integrar e coordenar os serviços disponibilizados por dispositivos com comunicação sem fio e onde as restrições de negócios são fornecidas pela aplicação comercial SAP. Em um trabalho complementar, (PUTTONEN *et al.*, 2008) apresentam uma arquitetura para a execução da coordenação de serviços em um ambiente industrial baseado na definição de processos padronizados via BPEL.

Em (PHAITHOONBUATHONG *et al.*, 2010) é apresentada uma arquitetura multicamada com quatro níveis para apresentar os serviços dos dispositivos de controle (sensores, atuadores e de realização) numa estrutura de rede e uma camada de interação utilizando o padrão DPWS.

Em (GARCIA MELO *et al.*, 2010), os autores apresentam uma estrutura multicamada de três níveis que disponibiliza as funcionalidades e/ou serviços de um SP disperso: (1) nível inferior que agrupa os serviços básicos envolvidos no processo de negócio dos SPs que compõem o SP disperso; (2) nível intermediário, onde se tem o coordenador que gerencia a execução dos serviços básicos e; (3) nível superior, onde se localizam os WSs que disponibilizam as funcionalidades dos SPs ao usuário (ou outros sistemas).

Ainda (GARCIA MELO, 2011) propôs uma arquitetura de um Sistema Coordenado de Teleoperação de Sistemas Produtivos (SCTSP) que adota a abordagem SOA para assegurar um ambiente de coordenação na execução das atividades dos SPs. A estrutura proposta é multicamada de três níveis. (1) nível

superior ou nível 1, onde um serviço expõe a funcionalidade do SCTSP em uma rede de comunicação. Adicionalmente, neste nível são incorporados componentes que permitem a consolidação do atendimento de pedidos (de fabricação de produtos) considerando a limitação de recursos disponíveis; (2) nível intermediário ou nível 2, onde são abordados mecanismos de integração e coordenação dos serviços dos SPs envolvidos no processo produtivo de produtos. Desta forma, o coordenador garante a integração entre os serviços dos SPs para o atendimento dos pedidos; (3) nível inferior ou nível 3, onde são tratados os serviços dos SPs que podem ser requisitados para o atendimento dos pedidos. Considera-se também a possibilidade de reconfiguração da estrutura de todos os SPs que podem ser envolvidos no processo produtivo do SCTSP. Essa reconfiguração permite tanto expor a funcionalidade de negócio como as funcionalidades que definem as operações produtivas dos SPs como serviços. Desta forma, é possível a interação do serviço de negócio com serviços de sistemas mais complexos, como o SCTSP. Adicionalmente, a requisição do serviço operativo está associada a operadores que atuam em diferentes localidades físicas. Também, em seu trabalho, (GARCIA MELO, 2011), propôs um procedimento estruturado de forma que se assegure uma adequada especificação da exposição de um SP como um serviço e a coordenação e integração deles para obter um produto final.

O projeto IMC-AESOP é um projeto europeu de pesquisa e desenvolvimento que faz parte da iniciativa conhecida como “*7th Framework Programme da União Europeia*”. Este projeto tem por objetivo estudar a abordagem SOA para a próxima geração de sistemas SCADA (*Supervisory Control and Data Acquisition*) e DCS (*Distributed Control Systems*), principalmente em aplicações de controle de processos. Este projeto iniciou-se em setembro de 2010 e terá duração de 30 meses, isto é, com finalização prevista em 2013. Alguns trabalhos iniciais já foram divulgados como o de (MINOR, 2012) que visa integrar os padrões OPC UA e DPWS, utilizando suas vantagens principais a fim de aplicá-los, por meio da SOA, na monitoração e controle de processos em plantas e em sistemas de manufatura. Em (NAGORNY, COLOMBO e SCHIMIDTMANN, 2012) é apresentada uma arquitetura de automação da manufatura orientada a serviços e multi-agentes compatível com o nível 2 da norma IEC 62264.

A Tabela 2.1 apresenta uma síntese de alguns dos diferentes trabalhos de SOA aplicados de alguma forma à área de SPs, na qual se pode observar que foram encontrados poucos trabalhos exploram a utilização de uma ferramenta formal para a especificação dos serviços da SOA e a comunicação entre estes.

Tabela 2.1 - Síntese de aspectos relevantes dos trabalhos de SOA aplicados a área de SPs.

Adaptado de (GARCIA MELO, 2011)

Autores	Tipo de sistema produtivo	Modelagem formal	Especificações adicionais
HOLMQVIST; PESSI (2004)	Sistema de fornecimento de partes automotivas	Não	Não
YAN; LIANG; DU (2005)	Laboratórios remotos	Não	<i>GRID</i>
MADHUSUDAN (2005)	Sistema de fornecimento de peças	Não	Não
KUMAR <i>et al.</i> (2005)	Sistema de veículo guiado automaticamente	Não	<i>Web Services Resource Framework</i>
SUA; GUOA; CHANGB (2005)	Fabrica de fundição	Não	<i>Dynamic Manufacturing Service Provision Mechanism</i>
MENG; YEXIN (2005)	Sistema de manufatura de peças	Não	<i>GRID</i>
LIU <i>et al.</i> (2006)	Sistema de manufatura de peças	Não	<i>Web Service Resource Framework</i>
KIM <i>et al.</i> (2006)	Sistema geral de manufatura	Não	<i>Web Service Choreography Interface</i>
LI-FENG; ZHOU (2006)	Sistema flexível de manufatura	Não	<i>Manufacturing Message Specification</i>
KOMODA (2006)	Sistema de manufatura de semicondutores	Não	Não
DELAMER; LASTRA (2006)	Sistema de montagem	Não	<i>Web Ontology Language for service</i>
BINDIETRICH; KIRN; SUGUMARAN (2007)	Sistema de manufatura de calçado	Não	Não
ZEEB <i>et al.</i> (2007)	Sistema de dispositivos embarcados	Não	<i>Devices Profile for Web Services</i>
CIANCETTA; D'APICE; GALLO (2007)	Sistema de sensores	Não	<i>Dynamic Host Configuration Protocol</i>
MENDES <i>et al.</i> (2007)	Sistema de robótico de transporte	Sim	<i>Devices Profile for Web Services</i>

Autores	Tipo de sistema produtivo	Modelagem formal	Especificações adicionais
KIRKHAM <i>et al.</i> (2008)	Sistema de montagem	Não	<i>Devices Profile for Web Services</i>
MOREIRA <i>et al.</i> (2008)	Sistema de manufatura	Não	<i>Devices Profile for Web Services</i>
MENDES <i>et al.</i> (2008)	Sistema de montagem	Sim	<i>Devices Profile for Web Services</i>
DOMNIC; STAMATIS <i>et al.</i> (2008)	Sistema de sensores sim fio	Não	<i>Devices Profile for Web Services</i>
GARCIA MELO <i>et al.</i> (2010)	Sistema de montagem	Sim	Não
PHAITHOONBUATHONG <i>et al.</i> (2010)	Sistema de montagem	Não	<i>Dynamic Host Configuration Protocol</i>
GARCIA MELO (2011)	Sistema de montagem	Sim	Não

2.2. REDE DE PETRI

A rede de Petri (RdP) é um meio formal, gráfico e executável para modelagem, especificação e análise de sistemas concorrentes e com uma dinâmica orientada por estados discretos e eventos instantâneos, conhecidos por sistemas a eventos discretos (SED) (ZHOU e VENKATESH, 1999), (NASSAR *et al.*, 2008), (ADAM, ATLURI e HUANG, 1998) . Desde sua apresentação por Carl Adam Petri (BRAUER e REISIG, 2006), a RdP tem sido usada na modelagem e análise de diferentes tipos de aplicações tais como: protocolos distribuídos (KANESHIRO *et al.*, 2007), aplicações industriais (NASSAR, MELO, *et al.*, 2008), tratamento de falhas (MORALES, MIYAGI e GARCIA MELO, 2007), análise do fluxo de processos (KIEPUSZEWSKI e VAN DER AALST, 2003), controle supervisorio (LEE, ZHOU e HSU, 2005), (GARCIA MELO, MORALES e MIYAGI, 2008), entre outras. Nas últimas décadas, diferentes extensões têm sido introduzidas explorando técnicas de interpretação, hierarquização e, modularização, entre outras (HASEGAWA, MIYAGI e TAKAHASHI, 1989) (LIU, MIYAGI e SCHREEK, 1994) (OYANG *et al.*, 2007) para

facilitar a modelagem e análise de sistemas complexos onde a implementação ou a coordenação na distribuição dos componentes e atividades são fatores importantes.

Segundo (VAN DER AALST, 1998), a RdP tem as seguintes vantagens na modelagem de fluxo de processos:

- Semântica formal que permite maior clareza e precisão na representação de fluxo dos processos no modelo. Isto é um dos fatores que tem motivado o estudo da semântica das RdPs.
- Representação gráfica que associada a semântica formal facilita a compreensão do modelo e a comunicação entre o grupo de pessoas com interesse no sistema abordado.
- Expressão de forma relativamente natural das especificações de fluxos de processo em SEDs, tais como: concorrência, conflito, sequência, sincronização, assincronismo, ciclos, entre outras.
- Disponibilidade de informação derivada do vasto material de divulgação em livros e artigos que têm sido gerados pela comunidade científica.
- Análise de processos e sistemas por meio de diferentes técnicas para provar as propriedades do sistema (segurança, vivacidade, limitação, etc.) e calcular o seu desempenho (tempo de resposta, de espera, de ocupação, etc.). Desta forma, é possível especificar, verificar e validar diferentes estratégias de controle de fluxo de processos.
- Independência de plataforma de *hardware* ou *software* já que a modelagem e análise na RdP não é baseada em aplicações computacionais disponibilizadas por um fornecedor específico.

De acordo com (HASEGAWA, 1996) e (ZHOU e VENKATESH, 1999), além das vantagens apresentadas anteriormente, outras características devem ser exploradas em aplicações de controle, tais como:

- Abstração: permite a identificação de estados e ações de modo claro e explícito, sendo efetivo para monitoração do sistema em tempo real.
- Procedimentos de modelagem: pode incorporar procedimentos do tipo refinamento (*top-down*) e do tipo composição modular (*bottom-up*) e/ou híbrido por meio de técnicas como: reutilização e decomposição.

2.2.1. Definição da Rede de Petri

A RdP é definida como um grafo bipartido composto por *lugares* (representados por círculos), e *transições* (representados por barras). Estes elementos são conectados mediante *arcos* orientados. É importante notar que ligações diretas entre elementos do mesmo tipo não são válidas (ROZENBERG e ENGELFRIET, 1996)

Assim, de acordo com (ZHOU e LI, 2008) uma rede de Petri é definida pela quádrupla (P, T, F, W) , onde:

P : é o conjunto finito de *lugares* (neste capítulo, a referência a elementos específicos da RdP são apresentadas neste texto em *itálico*)

T : é o conjunto finito de *transições* ($P \cap T = \emptyset$),

$F \subseteq (P \times T) \cup (T \times P)$ é o conjunto finito de *arcos* orientados

$W: F \rightarrow \mathbb{N}^+$ é o mapeamento que determina o peso de um arco.

Um *lugar* p é chamado de *lugar* de entrada da *transição* t se existe um *arco* orientado de p a t . O conjunto de *lugares* de entrada de t é denotado como $\bullet t$. Um *lugar* é chamado de *lugar* de saída da *transição* t se existe um *arco* orientado de t a p . O conjunto de *lugares* de saída de t é denotado como $t \bullet$. Analogamente, a notação $\bullet p$ e $p \bullet$ tem uma interpretação similar, isto é, $p \bullet$ é o conjunto de *transições* que compartilham um *lugar* p como *lugar* de entrada.

Considera-se também que os *lugares* podem conter um número inteiro e positivo de *marcas*, os quais são representados por pontos pretos em número equivalente à *marcação*. O estado do sistema é referido como a *marcação* M do sistema, ou seja, a distribuição de *marcas* nos *lugares*. Desta forma, $M = (1, 2, 0, 1)$ indica que no estado atual o sistema tem uma *marca* no *lugar* p_1 , duas *marcas* no *lugar* p_2 , não têm *marcas* no *lugar* p_3 e apresenta uma *marca* no *lugar* p_4 .

Para dois estados M_1 e M_2 , $M_1 \leq M_2$, se para todo $p \in P$: $M_1(p) \leq M_2(p)$.

A princípio, assume-se que os *lugares* têm capacidade infinita de receber *marcas*.

Considerando as *transições* como os elementos ativos da RdP que definem as mudanças no estado da rede, estabelece-se que as condições de *disparo* das *transições* são:

- 1- Uma *transição* t é dita *habilitada* se cada *lugar* de entrada de t tem um número de *marcas* maior ou igual ao número de *arcos* orientados do respectivo *lugar* para t . Este número de *arcos* é chamado de *peso do arco*.
- 2- Uma *transição habilitada* t pode ser *disparada* e, quando isto ocorre, as *marcas* nos *lugares* de entrada de t são reduzidas em um número equivalente ao *peso* dos respectivos *arcos* orientados para t e as *marcas* nos *lugares* de saída de t são acrescidas em um número equivalente ao *peso* nos respectivos *arcos* orientados.

Para uma RdP, $N = (P, T, F)$, e um estado M_1 , tem-se a notação a seguir:

$M_1[t]M_2$: A *transição* t está habilitada no estado M_1 e, quando ela é *disparada*, o estado da RdP evolui para M_2 .

$M_1 \rightarrow M_2$: faz referência a evolução do estado M_1 para o estado M_2 .

$M_1[\sigma]M_n$: representa uma sequência de disparos $\sigma = t_1 t_2 t_3 \dots t_{n-1}$ tal que o estado evolui de M_1 a M_n

Neste sentido, um estado M_n é alcançável a partir de M_1 (notação $M_1 \rightarrow M_n$) se existe uma sequência de *disparos* σ , tal que $M_1[\sigma]M_n$.

Uma RdP com um estado inicial M_0 é representada da forma (N, M_0) .

Análise da rede de Petri

Uma vantagem de aplicar a RdP ao estudo de SEDs é a sua formalização matemática que permite a verificação das propriedades de um sistema modelado (CARDOSO; VALETTE, 1997). Uma vez que o modelo em RdP é construído, este pode ser analisado de duas formas:

- **Análise estrutural:** onde se valida a lógica do modelo. Esta é uma análise estática com base na descrição estrutural do sistema modelado, dos dados e das regras que manipulam os dados e executam as tarefas. Assim, verifica-se a combinação de regras, dados e a estrutura que representam as entidades do sistema, isto é, verifica-se a composição das partes e se as relações entre elas são consistentes e completas.
- **Análise comportamental:** onde se examina o comportamento dinâmico do modelo, isto é, sua funcionalidade. Isso permite verificar se o modelo produz todas as saídas devidas para um dado estímulo, se o seu comportamento atende aos requisitos do usuário, etc.

Para avaliar o comportamento de RdP, existem muitas técnicas, tais como: a simulação e a análise do espaço de estados (GARCIA MELO, 2011).

Segundo (ZURAWSKI e ZHOU, 1994) a análise das propriedades de um sistema modelado em RdP pode ser realizada por meio de quatro abordagens principais: geração da árvore de alcançabilidade, análise da equação de estado, técnicas de redução e, simulação.

Interfaces entre modelos em Rede de Petri

Pode-se modelar um sistema em RdP a partir da modelagem de seus subsistemas também em RdP, sendo necessário a posterior composição destes modelos de RdP para se obter o modelo de todo o sistema. Em (GOMES e BARROS, 2005) apresenta-se vários trabalhos agrupados de acordo com a abordagem adotada para a composição de modelos em RdP. Duas das soluções apresentadas são a fusão de *lugares* e a fusão de *transições*. Desta forma, as interfaces nos modelos são “coladas” em um ponto específico (*lugar* ou *transição*). Alternativamente, outras abordagens fazem uso de extensões na definição da RdP. Assim, alguns trabalhos têm utilizado *arcos* habilitadores, e outros elementos que adicionam/manipulam informação nas *marcas*.

Em (GARCIA MELO, 2011) foi proposta uma notação que permite explicar o processo da composição dos modelos dos componentes de uma arquitetura distribuída, a qual é utilizada para que as formas de comunicação entre modelos em RdP possam ser expostas e comparadas. Esta notação é definida por:

- *modelo requisitante* – modelo em RdP que, por meio de uma interface (*lugar*, *transição* ou *arco* habilitador), requisita a funcionalidade de um segundo modelo em RdP;
- *modelo requisitado* – modelo em RdP que, por meio de uma interface (*lugar*, *transição* ou *arco* habilitador), disponibiliza uma funcionalidade para que outros modelos em RdP possam utilizá-la;
- *interface requisitante* ($\bullet$) – interface utilizada para chamar a funcionalidade desejada no modelo requisitado;
- *interface requisitada* ($\lrcorner$) – interface utilizada para receber solicitações de uma funcionalidade no modelo requisitado;
- *interface de resposta requisitante* ($\lrcorner$) – interface utilizada para receber a informação requisitada;
- *interface de resposta requisitada* ($\circ$) – interface utilizada para o envio da informação requisitada;

- *canal de comunicação* - conjunto composto por uma interface requisitante e uma interface requisitada. Este canal pode ser de requisição ou de resposta.

Assim, com o uso desta notação as diferentes instâncias dos canais de comunicação, requisitante e de resposta, que podem ser geradas (Figura 2.5

Figura 2.5 - a) Exemplo de modelo de fusão por *transições*, b) Modelo simplificado
(GARCIA MELO, 2011)

De acordo com a abordagem de composição por fusão de *transições*, no exemplo da Figura 2.5, as interfaces dos modelos requisitantes são explicitadas por meio das *transições* TA0 e TB0 e, as interfaces de resposta requisitante pelas *transições* TA1 e TB1. Analogamente, as interfaces requisitadas são explicitadas por meio das *transições* TC0T0 e TC0T e, as interfaces de resposta requisitada pelas *transições* TC2T0 e TC2T1. No modelo requisitado (Modelo C), são acrescentados *lugares* que representam a requisição sendo atendida (A ou B). Para a composição dos modelos, no envio da mensagem, são estabelecidos os canais de comunicação mediante a fusão de *transições*. Neste sentido o canal requisitante é definido pela fusão de *transições* de interface de resposta requisitante e interface de resposta requisitada. Assim, o canal requisitante de A para C (canal requisitante AC), é definido com a fusão das *transições* de interface TA0 e TC0T0, e o canal

requisitante de B para C (canal requisitante BC), é definido com a fusão das *transições* de interface TB0 e TC0T1. O canal de resposta do Modelo C para Modelo A (canal de resposta AC) é definido com a fusão das *transições* de interface TA1 e TC2T0 e o canal de resposta do Modelo C para Modelo B (canal de resposta BC) é definido com a fusão das *transições* de interface TB1 e TC2T1.

2.2.2. Grafo de Fluxo da Produção

O grafo de fluxo de produção, mais conhecido como *Production Flow Schrema* (PFS) (MIYAGI, 1996) é uma classe de rede canal-agente (REISIG, 1992) derivada da RdP devidamente interpretada e constitui uma técnica para desenvolver uma descrição de um sistema a partir de uma visão conceitual, ou seja, em um alto nível de abstração (SANTOS FILHO, 1998) sem o detalhamento do seu comportamento dinâmico.

O PFS caracteriza um sistema a partir de seus elementos ativos e o fluxo de materiais e informações dentro de uma visão de processo produtivo. Esta abordagem baseia-se no princípio de que um sistema é composto por elementos ativos que realizam atividades, isto é, transformações sobre itens (materiais ou informações) e elementos passivos que realizam o armazenamento e a distribuição destes, existindo, portanto, uma inter-relação entre estes elementos para a descrição efetiva dos fluxos presentes no sistema.

O PFS utiliza a técnica de refinamentos sucessivos por meio de uma abordagem *top-down*, de maneira que seja possível inserir progressivamente, no modelo, o detalhamento na descrição dos processos do sistema. O objetivo é representar o fluxo de operações tendo como referência a evolução dinâmica de um determinado processo. Deve-se salientar que o PFS, distintamente da RdP, não possui o conceito de *marcação*, pois é um modelo conceitual do sistema com o foco na descrição estrutural dos fluxos de itens e informações/dados no sistema.

O PFS é composto pelos seguintes elementos (Figura 2.6). Neste texto, as referencias dos elementos específicos do PFS são apresentadas em sublinhado.

1. Atividade: representa um componente ativo do sistema capaz de realizar transformações, isto é, ações. Graficamente, corresponde a um macro-elemento delimitado por colchetes e com inscrições em linguagem formal para especificá-lo. Cada atividade pode ser refinada em sub-atividades com as suas respectivas interações;
2. Inter-atividade ou distribuidor: representa um componente passivo, não realiza transformação, mas é capaz de armazenar e distribuir os itens. Graficamente é indicado por uma circunferência. Uma inter-atividade deve existir sempre entre duas atividades que estejam relacionadas;
3. Arco orientado: representa as relações entre as atividades e as inter-atividades. Os arcos indicam uma relação lógica, entre os componentes. Se a conexão se faz pela parte externa da atividade (conectado a um dos colchetes), indica um fluxo principal de itens no sistema e, se é realizada pela parte interna (entre os colchetes), indica um fluxo secundário (este fluxo não é obrigatório). Graficamente corresponde a uma seta.

Figura 2.6 – Componentes do PFS. (GARCIA MELO, 2011)

2.2.3. Procedimento PFS/RdP

Dentro das abordagens derivadas da RdP, destaca-se o PFS/RdP (*Production Flow Schema/Rede de Petri*) cujos princípios estão descritos em (MIYAGI, 1996) e que visa sistematizar a modelagem de SEDs segundo uma abordagem hierárquica, onde, com base em sucessivos refinamentos é construído um modelo de forma progressivo e estruturado de modo que são preservadas as estruturas das

atividades do nível superior e são também descritos os seus detalhes operacionais num nível funcional. A ideia é que, se um conjunto de regras é seguido na construção da RdP, pode-se assegurar que este intrinsecamente apresenta propriedades desejáveis como vivacidade, segurança, reversibilidade, etc. (MORAES e CASTRUCCI, 2007) . Inicialmente o modelo conceitual do sistema (representando um alto nível de abstração do sistema sem detalhamento de sua dinâmica) é desenvolvido com PFS. Nesta etapa o propósito é modelar as principais características estruturais que serão consideradas no sistema. A ênfase está na identificação dos componentes ativos e passivos do sistema, assim como do fluxo de itens (material e/ou informação) entre elementos. Um componente ativo é quem realiza uma atividade, um componente passivo é um distribuidor de itens. De acordo com a metodologia, os elementos do PFS são detalhados gradativamente segundo uma estrutura hierárquica. Este detalhamento pode gerar três tipos de sub-grafos: totalmente em PFS; com alguns elementos em PFS e outros em RdP; totalmente em RdP.

Na modelagem do comportamento dinâmico do sistema, o modelo em PFS é convertido progressivamente em um modelo em RdP que detalha o funcionamento das diversas partes do sistema (até o nível desejado) (MIYAGI, 1996).

O PFS/RdP como técnica para modelar SEDs, originalmente foi proposta para aplicação em sistemas de manufatura e, neste campo tem sido usada para tratar sua modelagem e análise com sucesso (RIASCOS e MIYAGI, 2001) (NASSAR *et al.*, 2008).

2.3. SÍNTESE DO CAPÍTULO

Neste capítulo apresentou-se os aspectos básicos para a implementação da SOA, como o WS, WSDL, aplicações de SOA em SPs, RdP e PFS.

O levantamento realizado indica que apesar de existirem várias propostas de aplicação do SOA em SPs, somente em (GARCIA MELO, 2011) se propôs um procedimento sistematizado para o projeto e implementação da solução de orquestração dos serviços, em especial no caso de SPs dispersos. Porém, conforme

identificado no mesmo trabalho, existe a necessidade de que em SP disperso a solução de orquestração deve ter capacidade de incluir SPs de acordo com a funcionalidade requerida segundo critérios pré-determinados, sendo esta solução a ser apresentada nesta dissertação.

Neste capítulo também foram abordados os conceitos básicos de uma ferramenta formal de modelagem e análise como é a RdP, e do procedimento PFS/RdP. Estas técnicas são consideradas fundamentais para a modelagem e análise dos processos num SP disperso e, no presente trabalho são exploradas também para a geração de uma especificação para a implementação da SOA para SPs dispersos.

Assim, no Capítulo 3 apresenta-se a proposta de uma arquitetura de controle de SP disperso baseada na SOA que pode ser avaliada e implementada de acordo com um procedimento sistematizado de projeto baseado no PFS/ RdP citado no Capítulo 4.

3. ARQUITETURA DE CONTROLE PARA ORQUESTRAÇÃO DE SPs DISPERSOS

Neste capítulo apresenta-se a arquitetura de controle para orquestração de SPs dispersos. Esta arquitetura visa integrar e coordenar o conjunto de SPs envolvidos para a realização dos processos produtivos. Conforme já descrito na introdução, assume-se que os SPs possuem suas funcionalidades disponibilizadas como serviços, isto é, existe uma aplicação que expõe as funcionalidades deste SP através de um serviço, no caso um WS. A arquitetura de controle para orquestração de SPs desenvolvida permite a inclusão dinâmica de SPs de acordo com políticas de prioridade que melhor atenderem os requisitos para a produção dos produtos.

3.1. FUNDAMENTOS DA ARQUITETURA PROPOSTA

Conforme já descrito, SPs são aqueles sistemas que utilizam recursos para produção de bens e serviços. Arquitetura é a estrutura lógica que fornece suporte aos elementos do sistema. Por sua vez, orquestração é aqui entendida como uma abordagem centralizada para controle de um grupo de lógicas de processo que permite a interoperabilidade entre duas ou mais aplicações (CÂNDIDO, BARATA, *et al.*, 2009b). Conforme o levantamento realizado, já existem algumas propostas de arquiteturas para controle de SPs baseadas na SOA, porém em nenhum dos casos se identificou uma ferramenta formal para a validação da arquitetura. Em todos os trabalhos a arquitetura adotada é desenvolvida de forma *ad-hoc* e assim apesar de indicarem desempenhos adequados para os casos considerados a generalização dos resultados obtidos para outras situações não é aplicável. Somente em (GARCIA MELO, 2011) é que se utilizou uma ferramenta formal para a validação da arquitetura, além de um procedimento estruturado para o desenvolvimento do Sistema de Coordenação e Teleoperação de Sistemas Produtivos (SCTSP). Baseado no porte e complexidade de SPs dispersos reais entende-se que

a abordagem de (GARCIA MELO, 2011) pode e deve ser explorada para a especificação da arquitetura de controle de SPs dispersos.

A arquitetura apresentada em (GARCIA MELO, 2011), pode ser observada na Figura 3.1. Esta arquitetura possui três níveis: nível 1 responsável pelo gerenciamento do pedido realizado pelo ator “Cliente” do SCTSP; nível 2, em que ocorre a integração e coordenação dos SPs e; nível 3, composto pelos SPs envolvidos no processo produtivo do produto requisitado pelo “Cliente”. Particularmente, o “Serviço de Integração e Coordenação”, situado no nível 2 da arquitetura, é o “orquestrador” do SCTSP, responsável por coordenar de maneira centralizada os processos produtivos globais. Este serviço é o que integra os diferentes SPs envolvidos na execução dos processos. Para isso, os SPs coordenados possuem suas funcionalidades expostas como serviços, como por exemplo, em “Serviço do Sistema Teleoperativo de Produção 1”, conforme ilustrado na Figura 3.1. Porém, a limitação nesta arquitetura é de que o “*Serviço de Integração e Coordenação*” é capaz apenas de tratar sequencialmente os SPs que foram especificados previamente. Dessa maneira, a arquitetura não prevê nenhum mecanismo de inserção de novos SPs ou de seleção de SPs com as mesmas funcionalidades e diferentes desempenhos. Também, outra limitação é que caso algum SP do sistema global (SP disperso) se torne indisponível, o processo produtivo global é interrompido até que se tenha novamente a disponibilidade deste SP. Desta forma, se introduz aqui uma solução para orquestração de SPs, que permite realizar esta inserção e seleção dinâmica de SPs. O foco aqui está na reconfiguração do nível 2 (Figura 3.2), com uma nova proposta para orquestração de SPs dispersos, de maneira a tratar de casos como a inserção e seleção de SPs. Esta orquestração de SPs dispersos foi desenvolvida com o uso de um procedimento estruturado, em que se levantou os casos de uso, e por meio da modelagem PFS/RdP se analisou os modelos resultantes dos processos da arquitetura, por meio da análise estrutural (lógica do modelo) e comportamental (através de simulação dos modelos em RdP). Estes modelos representam a interação entre os serviços e componentes da arquitetura de orquestração de SPs dispersos proposta.

Figura 3.1 - Arquitetura proposta em (GARCIA MELO, 2011)

Figura 3.2 - Arquitetura proposta neste trabalho, com a reconfiguração do nível 2

3.2. DESCRIÇÃO GERAL DA ARQUITETURA PROPOSTA

A reconfiguração citada resulta na concepção de uma nova arquitetura para orquestração de SPs dispersos que pode ser observada na Figura 3.3. A orquestração de SPs aqui proposta é composta por dois serviços principais: o “*Serviço de Coordenação*” e o “*Serviço de Integração e Descoberta*”. O “*Serviço de Coordenação*” é responsável por realizar a coordenação dos diferentes SPs que compõem o SP disperso para o atendimento dos pedidos. O “*Serviço de Integração e Descoberta*” é responsável pela seleção e inserção dos SPs que serão utilizados para atender ao pedido.

Figura 3.3 - Arquitetura para orquestração de SPs dispersos

Considera-se neste trabalho que o processo produtivo global para a produção de um determinado produto é realizado por um conjunto coordenado de processos produtivos de funcionalidades específicas, conforme indicado na Figura 3.4. Além disso, SPs diferentes podem apresentar a mesma funcionalidade para a produção de um produto, bem como os SPs também podem possuir mais de uma funcionalidade diferente. O termo “pedido” neste texto é utilizado para indicar a solicitação por produtos ao sistema de controle de SPs dispersos e, que, por meio do qual se inicia o processo produtivo global.

Figura 3.4 - Processo produtivo global

A seguir é explicado cada um dos componentes necessários para efetivamente realizar a orquestração de SPs dispersos.

3.2.1. Integrador de SPs

O “*Integrador de SPs*” é o ator responsável por realizar a inserção de novos SPs no sistema através da “*Interface de Integração*”. Neste caso, o ator pode ser um usuário do sistema de controle de SPs ou uma aplicação computacional desenvolvida para esta função. O novo SP é cadastrado de acordo com a sua funcionalidade oferecida, exposta como serviço. A “*Interface de Integração*” é a interface de interação entre o “*Integrador de SP*” e o “*Serviço de Integração e Descoberta*”. Como os serviços são implementados com WS, esta “*Interface de Integração*” pode ser acessada por meio de um *web site*, por exemplo, entre outros.

3.2.2. Serviço de Integração e Descoberta

O “*Serviço de Integração e Descoberta*” possui duas finalidades principais: a integração de novos SPs e a busca e Descoberta de SPs para atender ao pedido. Estes termos “*integração*” e “*descoberta*” foram selecionados para indicar e descrever de maneira clara o papel deste componente (“*Serviço de Integração e*

Descoberta”) na arquitetura do sistema de controle para orquestração de SPs. Baseando-se no próprio conceito de SOA e WS, este serviço centraliza e armazena em um banco de dados de SPs (*“Diretório Ativo de Serviços dos SPs”*, Figura 3.2) todas as informações referentes aos SPs existentes, classificados de acordo com sua funcionalidade. Para a descrição deste componente, realizada a seguir, serão utilizados aspectos conceituais e de implementação para melhor entendimento da finalidade do *“Serviço de Integração e Descoberta”* na arquitetura:

a) *Integração de novos SPs / Comparador de WSDL*

Ao receber uma requisição para o cadastro de um “novo SP” (exposto como serviço), o *“Serviço de Integração e Descoberta”* utiliza o componente *“Comparador de WSDL”* para comparar o WSDL do serviço do “novo SP” com o WSDL padrão para os serviços que apresentam a mesma funcionalidade. O *“Comparador de WSDL”* compara assim a descrição das operações (ou métodos) descritos no WSDL do serviço do “novo SP” a ser integrado, com os métodos existentes no WSDL padrão do serviço de SP desta determinada funcionalidade, a fim de se verificar se elas são condizentes com a sintaxe necessária para efetuar a correta invocação. Também, existe a possibilidade de um SP ser capaz de ter mais de uma funcionalidade diferente, sendo que o *“Comparador de WSDL”* busca apenas os métodos necessários que condizem com determinada funcionalidade em questão.

Dessa maneira, se o *“Serviço de Integração e Descoberta”*, por meio da comparação de WSDLs, determinar que a interface do serviço do “novo SP” atende as especificações para comunicação com os demais serviços da arquitetura, este “novo SP” é cadastrado no banco de dados de SPs e fica disponível para uso.

b) *Partner Selection* : Mecanismo de Seleção/Agente Descobridor

O *“Serviço de Integração e Descoberta”* ao receber uma solicitação do *“Serviço de Coordenação”* deve selecionar quais os SPs, de cada funcionalidade diferente, devem ser responsáveis pelo atendimento do pedido. O elemento *“Mecanismo de Seleção”*, indicado na Figura 3.2, é responsável por realizar esta seleção. O *“Mecanismo de Seleção”* solicita ao

“*Agente Descobridor*” a lista de SPs a serem selecionados para cada funcionalidade diferente. Dessa maneira, o “*Agente Descobridor*” realiza a ligação entre o “*Mecanismo de Seleção*” e o “*Diretório Ativo de Serviços*”. Observa-se que o foco deste trabalho não é definir ou propor novos mecanismos de seleção de parceiros (*Partner Selection*) (BALDO, RABELO e VALLEJOS, 2009), (WU e SU, 2005) ou algoritmos para *scheduling* de pedidos (PESSOA *et al.*, 2012), e sim fornecer uma arquitetura para orquestração de SPs dispersos. Por esta razão, tem-se este elemento (“*Mecanismo de Seleção*”) independente para realizar a seleção ou a comunicação com outros sistemas independentes para se realizar a seleção. Por fim, após a seleção dos SPs, o “*Serviço de Coordenação*” recebe a lista de SPs selecionados para cada funcionalidade.

Durante a execução do processo produtivo global, um SP previamente selecionado pode se tornar indisponível. Para que não seja necessário aguardar a disponibilidade deste SP ou até mesmo interromper a execução do pedido, utiliza-se o elemento “*Agente Descobridor*” para selecionar um outro SP e, assim, substituir o SP indisponível para esta funcionalidade. Este SP é selecionado de acordo com o critério prioritário (por exemplo, menor custo ou maior qualidade de serviço) escolhido pelo cliente, durante a solicitação do novo pedido.

c) Atualização de Estados de SPs

Para o correto funcionamento da orquestração de SPs, é necessário que o banco de dados de SPs (i.e., “*Diretório ativo de serviços de SPs*”) possua informações atualizadas sobre estes. Para isso, baseou-se no conceito utilizado para dispositivos por meio do DPWS (SOCRADES, 2008) e pela especificação WS-Discovery (OASIS, 2009), em que, a fim de minimizar a necessidade de *polling*, cada serviço, ao ser conectado, e que deseja ser descoberto, envia um “anúncio” toda vez que pretende fazer parte ou se retirar da rede. Dessa forma, toda vez que um SP alterar sua disponibilidade ou alguma característica que está sendo monitorada, ele deve invocar o “*Serviço de Integração e Descoberta*” para informar o seu novo *status*, a fim

de que esse atualize o banco de dados de SPs, através do método *“Atualiza Estados”*.

3.2.3. Cliente / Interface Cliente

O *“Cliente”* é o ator responsável por introduzir um novo pedido de produto ao sistema de controle de SPs, a fim de que seja realizada a orquestração. O *“Cliente”* ao realizar um novo pedido de determinado produto, deve também selecionar qual o critério de prioridade para ser utilizado, em caso de indisponibilidade de algum SP, na seleção de um novo SP. A *“Interface Cliente”* deve apresentar quais são os critérios existentes para escolha, como por exemplo, menor custo, menor prazo de entrega, maior qualidade de serviço, entre outros que serão armazenados no *“Banco de Dados de Pedidos”* na realização do pedido. É importante frisar que o ator *“Cliente”* e que a *“Interface Cliente”* tratam apenas exemplos de aplicação e têm a função de introduzir o pedido ao sistema de controle para orquestração de SPs dispersos. O gerenciamento de pedidos, *scheduling* de pedidos, e as demais camadas de negócios de uma empresa (*virtual* ou não), não é o foco neste trabalho, sendo possível substituir esta camada por qualquer outra aplicação do nível de negócios (SAP, por exemplo) que forneça ao *“Serviço de Coordenação”* as informações necessárias para iniciar o pedido.

3.2.4. Serviço de Coordenação

O *“Serviço de Coordenação”* é o serviço responsável por coordenar os SPs de um determinado processo produtivo para atender a um pedido. Este serviço centraliza o *know-how* do processo produtivo global, sabendo quais são as funcionalidades (de SPs) necessárias e qual a sequência de SPs a serem convocadas/reservadas para a fabricação de um determinado produto. O *“Serviço de Coordenação”* recebe as informações sobre o pedido (produto, quantidade, critério prioritário, etc.) do *“Cliente”* por meio da *“Interface do Cliente”*. Com isso, o *“Serviço de Coordenação”* solicita a lista de SPs selecionados ao *“Serviço de Integração e Descoberta”* e, para cada funcionalidade necessária ao atendimento do

pedido, seleciona os SPs de acordo com o seu “*Mecanismo de Seleção*”. Com a resposta proveniente do “*Serviço de Integração e Descoberta*”, o “*Serviço de Coordenação*” requisita somente aos serviços dos SPs selecionados, a fim de realizar a coordenação das tarefas e obter o produto final. Durante a execução do pedido, caso algum SP se torne indisponível, o “*Serviço de Coordenação*” utiliza o “*Serviço de Integração e Descoberta*” para solicitar outro SP, de acordo com o critério prioritário cadastrado.

Ao se concentrar o *know-how* do processo produtivo global em um único componente, o “*Serviço de Coordenação*”, durante a inserção ou alteração de um pedido de produto, a reconfiguração das etapas do processo produtivo global deve ser feita em um único componente da arquitetura, sem a necessidade de nenhuma modificação conceitual ou estrutural da arquitetura.

3.3. DIAGRAMAS DE CASOS DE USOS

Os diagramas de casos de uso com os requisitos necessários para o “*Serviço de Coordenação*” e para o “*Serviço de Integração e Descoberta*” encontram-se na Figura 3.5 e Figura 3.6, respectivamente.

Figura 3.5 - Diagrama de casos de uso do "Serviço de Coordenação"

Figura 3.6 - Diagrama de casos de uso do "Serviço de Integração e Descoberta"

3.4. SÍNTESE DO CAPÍTULO

Este capítulo apresentou a estrutura geral da arquitetura de controle para a orquestração de SPs dispersos. Os componentes fundamentais são explicados, assim como os principais serviços.

Entretanto, para a implementação desta arquitetura é necessário uma especificação funcional de cada um dos componentes e serviços. Desta forma o capítulo seguinte apresenta como estas especificações são geradas de acordo com um procedimento sistematizado baseado no PFS/RdP.

4. ESPECIFICAÇÃO DOS COMPONENTES/SERVIÇOS DA ARQUITETURA PROPOSTA

Este capítulo apresenta como o procedimento PFS/RdP é aplicado para modelagem do “*Serviço de Coordenação*” e do “*Serviço de Integração e Descoberta*”. Os modelos gerados, depois de uma verificação e validação, são assumidos como a especificação para a implementação dos componentes da arquitetura proposta.

4.1. SERVIÇO DE INTEGRAÇÃO E DESCOBERTA

De acordo como o diagrama de caso de uso do “*Serviço de Integração e Descoberta*” apresentado anteriormente, apresenta-se a seguir a modelagem em PFS e depois o seu refinamento em RdP de cada uma das funções identificadas.

4.1.1. Atividades de [Cadastro de SPs] e [Funções de Cadastro]

O refinamento das atividades [Cadastro de SPs] e [Funções de Cadastro] em PFS está ilustrado na Figura 4.1. Neste refinamento são apresentadas as atividades necessárias para o cadastro de um novo SP. O ator “*Integrador de SPs*”, através da “*Interface de Integração*”, realiza a atividade modelada em [Funções de Cadastro]. No refinamento da atividade [Cadastro de SPs], encontra-se a atividade [Verificação da Condição de Cadastro SP] que modela a verificação do WSDL do serviço que expõe o SP a ser cadastrado e que compara se, para a determinada funcionalidade, este serviço possui interface de acordo com os requisitos para o cadastro. A atividade [Notificação de Cadastro] informa a [Resposta Cadastro SP] se o SP foi ou não cadastrado.

Figura 4.1 - Refinamento das atividades [Cadastro de SPs] e [Funções de Cadastro SP]

Os modelos funcionais em RdP das atividades [Cadastro de SPs] e [Funções de Cadastro] são apresentados na Figura 4.2. A descrição dos elementos dos modelos em RdP (*lugares* e *transições*) que representam as operações estão nas Tabela 4.1 e Tabela 4.2.

Figura 4.2 – Modelo em RdP das atividades [Cadastro de SPs] e [Funções de Cadastro SP]

Considera-se, no modelo da Figura 4.2, que existe fusão de transição entre as transições T1a e T1b, T3a e T2b, e por fim, T4a e T3b.

Tabela 4.1 - Descrição dos elementos do modelo de RdP para a atividade [Cadastro de SP]

<u>Atividade</u> [Cadastro de SP]	
Lugar	Descrição
L1a	Aguarda uma mensagem de requisição para cadastro de SP
L2a	Mensagem de requisição de cadastro foi recebida
L3a	Preparação para verificação do serviço do SP a ser cadastrado
L4a	Verificação do serviço do SP foi realizada
L5a	Preparação para envio de resposta de não autorização de cadastro de SP
L6a	Resposta de não autorização de cadastro de SP enviada
L7a	Preparação para envio de resposta de autorização de cadastro de SP
L8a	Resposta de autorização de cadastro de SP enviada
Transição	Descrição
T1a	Transição requisitada para solicitar cadastro de SP
T2a	Transição interna de verificação de WSDL de serviço do SP a ser cadastrado
T3a	Transição de resposta requisitada não autorizando o cadastro do serviço do SP
T4a	Transição de resposta requisitada autorizando o cadastro do serviço do SP

Tabela 4.2 - Descrição dos elementos do modelo de RdP para atividade[Funções de Cadastro de SP]

<u>Atividade</u> [Funções Cadastro de SP]	
Lugar	Descrição
L1b	Preparação da mensagem para solicitar cadastro de SP

Lugar	Descrição
L2b	Mensagem de solicitação de cadastro de SP enviada
L3b	Aguarda mensagem sobre autorização de cadastro
L4b	Mensagem de não autorização de cadastro é recebida
L5b	Mensagem de autorização de cadastro é recebida
L6b	Resposta de cadastro concluída

Transição	Descrição
T1b	Transição requisitante de autorização de cadastro de SP
T2b	Transição de resposta requisitante não autorizando o cadastro do serviço do SP
T3b	Transição de resposta requisitante autorizando o cadastro do serviço do SP

Considera-se que existe fusão de transição entre as transições T1a e T1b, T3a e T2b, e por fim, T4a e T3b.

4.1.2. Atividades [Atualiza Estados] e [Funções de Atualização de Estados]

O refinamento das atividades [Atualiza Estados] e [Funções de Atualização de Estados] em PFS está na Figura 4.3. Neste refinamento são apresentadas as atividades necessárias para a atualização dos estados de um SP no banco de dados dos serviços dos SPs do sistema de orquestração. Cada um dos SPs, assim que tiver alguma alteração do seu estado (por exemplo, custo ou disponibilidade), deve atualizar o sistema com seu estado atual. Com isso o tráfego de informações na rede de comunicação será menor do que o tráfego que ocorreria se o sistema de orquestração realizasse a varredura e atualização de estados de todos os SPs cadastrados em determinados períodos de tempo.

Figura 4.3 - Refinamento do modelo em PFS das atividades [Atualiza Estados] e [Funções de Atualização de Estados]

Os modelos funcionais em RdP das atividades [Atualiza Estados] e [Funções de Atualização de Estado do SP] são apresentadas na Figura 4.4. A descrição dos elementos dos modelos em RdP (*lugares* e *transições*) que representam as operações encontram-se nas Tabela 4.3 e Tabela 4.4 .

Considera-se que no modelo da Figura 4.4 existe fusão de transição entre as transições T1a e T1b, e entre T3a e T2b.

Figura 4.4 – Modelo em RdP das atividades [Atualiza Estados] e [Funções de Atualização de Estado do SP]

Tabela 4.3 - Descrição dos elementos do modelo de RdP para a atividade [Atualiza Estados]

<u>Atividade</u> [Atualiza Estado]	
Lugar	Descrição
L1a	Aguarda uma mensagem de requisição para atualização de Estado
L2a	Mensagem de requisição de atualização foi recebida
L3a	Preparação para atualização do registro do estado do SP
L4a	Atualização do estado do SP foi registrada
L5a	Preparação para envio de mensagem de notificação de atualização do estado
L6a	Mensagem de notificação de atualização do estado foi enviada
Transição	Descrição
T1a	Transição requisitada para atualização de estado de SP
T2a	Transição interna de atualização do estado do SP
T3a	Transição de resposta requisitada de atualização de estado do SP

Tabela 4.4 - Descrição dos elementos do modelo de RdP para a atividade [Funções de Atualização de Estado do SP]

<u>Atividade</u> [Funções de Atualização de Estado]	
Lugar	Descrição
L1b	Preparação da mensagem para solicitar atualização de estado de SP
L2b	Mensagem de solicitação de atualização de estado de SP enviada
L3b	Aguarda mensagem de atualização de estado de SP
L4b	Mensagem de atualização de estado de SP foi recebida

Transição	Descrição
T1b	Transição requisitante de atualização de estado de SP
T2b	Transição de resposta requisitante de atualização de estado do SP

4.1.3. Atividades [*Solicitação de SPs*] e [*Funções de Solicitação de SPs*]

O refinamento das atividades [*Solicita SPs*] e [*Funções de Solicitação de SPs*] em PFS está descrito na Figura 4.5. Neste refinamento são apresentadas as atividades para a solicitação de SPs necessários para a realização de um determinado processo produtivo. É feita a requisição de uma lista de SPs para cada uma das funcionalidades, sendo que cada um dos itens desta lista contém informação sobre o endereço do serviço e as características principais sobre as quais se deseja fazer a seleção, preparados em formato de uma *string*. Após a solicitação da lista de SPs, para cada uma das funcionalidades diferentes de um processo produtivo, é realizada a “execução de seleção”, em que de acordo com o algoritmo para seleção de parceiros existente, é realizada a escolha da melhor configuração para atendimento do pedido.

Figura 4.5 - Refinamento do modelo em PFS para as atividades [Solicita SPs] e [Funções de Solicitação de SPs]

Os modelos funcionais em RdP das atividades [Solicita SPs] e [Funções de Solicitação de SPs] são apresentadas na Figura 4.6. A descrição dos elementos dos modelos em RdP (*lugares* e *transições*) que representam as operações encontram-se na Tabela 4.5 e Tabela 4.6.

Considera-se, no modelo da Figura 4.6, que existe fusão de transição entre as transições T1a e T1b, e entre T5a e T2b.

Figura 4.6 - Modelo em RdP das atividades [Solicita SPs] e [Funções de Solicitação de SPs]

Tabela 4.5 – Descrição dos elementos do modelo de RdP para as atividades [Solicita SPs]

Atividade [Solicita SPs]	
Lugar	Descrição
L1a	Aguarda uma mensagem de requisição para solicitação de SPs
L2a	Mensagem de requisição de solicitação foi recebida

Lugar	Descrição
L3a	Preparação para solicitação da lista de SP1 (de determinada funcionalidade)
L4a	Lista de SP1 foi recebida
L5a	Preparação para solicitação da lista de SP2 (de mesma funcionalidade)
L6a	Lista de SP2 foi recebida
Lna	Preparação para solicitação da lista de SPn (de mesma funcionalidade)
Lna	Lista de SPn foi recebida
L7a	Preparação para seleção de SPs para determinado pedido
L8a	SPs para determinado pedido selecionados
L9a	Preparação para envio da mensagem com lista de SPs selecionados
L10a	Mensagem com lista dos SPs selecionados enviada
Transição	Descrição
T1a	Transição requisitada para solicitação de SPs
T2a	Transição interna de requisição de lista de SP1s
T3a	Transição interna de requisição de lista de SP2s
Tna	Transição interna de requisição de lista de SPns
T4a	Transição interna de seleção de SPs
T5a	Transição de resposta de lista de SPs selecionados

Tabela 4.6 - Descrição dos elementos do modelo de RdP para a atividade [Funções de Solicitação de SPs]

<u>Atividade</u> [Funções de Solicitação de SPs]	
Lugar	Descrição
L1b	Preparação da mensagem para solicitar lista de SPs selecionados para determinado pedido
L2b	Mensagem de solicitação de lista de SPs enviada
L3b	Aguarda mensagem com lista de SPs selecionados
L4b	Mensagem com lista de SPs selecionados foi recebida
Transição	Descrição
T1b	Transição requisitante para seleção de lista de SPs
T2b	Transição de resposta requisitante com lista de SPs

4.1.4. Atividades [Solicita Único SP] e [Funções de Solicitação de único SP]

O refinamento das atividades [Solicita único SP] e [Funções de Solicitação de único SP] em PFS está descrito na Figura 4.7. Estas atividades são bastante similares às atividades [Solicita SPs] e [Funções de Solicitação de SPs]. A diferença consiste em que a atividade [Solicita único SP] é realizada quando, durante a execução de um processo produtivo, um SP de determinada funcionalidade previamente selecionado pela atividade [Solicita SPs] se torna indisponível. Assim, por meio da atividade [Solicita único SP] é possível selecionar um SP de determinada funcionalidade de acordo com o critério prioritário, previamente definido pelo “Cliente”.

Figura 4.7 - Refinamento do modelo em PFS das atividades [Solicita Único SP] e [Funções de Solicitação de único SP]

Os modelos funcionais em RdP das atividades [Solicita único SP] e [Funções de Solicitação de único SP] são apresentadas na Figura 4.8. A descrição dos elementos dos modelos em RdP (*lugares* e *transições*) que representam as operações encontram-se na Tabela 4.7 e Tabela 4.8.

Considera-se, no modelo da Figura 4.8, que existe fusão de transição entre as transições T1a e T1b, e entre T3a e T2b.

Figura 4.8 - Modelo em RdP das atividades *[Solicita único SP]* e *[Funções de Solicitação de único SP]*

Tabela 4.7 – Descrição dos elementos do modelo de RdP para as atividades *[Solicita único SP]*

Atividade <i>[Solicita único SP]</i>	
Lugar	Descrição
L1a	Aguarda uma mensagem de requisição para solicitação de SP específico
L2a	Mensagem de requisição de solicitação foi recebida

Lugar	Descrição
L3a	Preparação para seleção de SP de determinada funcionalidade, de acordo com o critério prioritário
L4a	SP selecionado
L5a	Preparação para envio da mensagem com SP selecionado
L6a	Mensagem com SP selecionado enviada
Transição	Descrição
T1a	Transição requisitada para solicitação de SP
T2a	Transição interna de seleção de SP
T3a	Transição de resposta de SP selecionado

Tabela 4.8 - Descrição dos elementos do modelo de RdP para as atividades [Funções de Solicitação de único SP]

<u>Atividade</u> [Funções de Solicitação de único SP]	
Lugar	Descrição
L1b	Preparação da mensagem para solicitar SP específico
L2b	Mensagem de solicitação de SP enviada
L3b	Aguarda mensagem com SP selecionado
L4b	Mensagem com SP selecionado foi recebida
Transição	Descrição
T1b	Transição requisitante para seleção de SP
T2b	Transição de resposta requisitante com SP selecionado

4.2. SERVIÇO DE COORDENAÇÃO

De acordo como o diagrama de caso de uso do “*Serviço de Coordenação*” apresentado anteriormente, apresenta-se a seguir a modelagem em PFS e depois o refinamento em RdP de cada uma das funções identificadas.

4.2.1. Atividades [*Novo Pedido*], [*Seleciona Critério*] e atividades [*Funções de Registro de Pedido*] e [*Funções de Registro de Critério para Pedido*]

O refinamento das atividades [*Novo Pedido*], [*Seleciona Critério*] e das respectivas atividades de [*Funções de Registro de Pedido*] e [*Funções de Registro de Critério para Pedido*] estão descritos nas Figura 4.9 e Figura 4.10. As atividades [*Função de Registro de Pedido*] e [*Funções de Registro de Critério para Pedido*] representam os modelos da interface que o cliente possui para fazer um novo pedido de produtos onde, para cada novo pedido, deve também selecionar o critério prioritário.

Ao iniciar a atividade [*Novo Pedido*] por meio da atividade [*Início Novo Pedido*] indicada é realizado o [*Cadastro do Pedido Novo*] no banco de dados de pedidos e, em seguida é feita a solicitação dos SPs a serem selecionados para este pedido, por meio da atividade [*Solicita SPs*] do “*Serviço de Integração e Descoberta*”, descrito anteriormente. Por fim, é realizada a notificação do ator “*Cliente*”, de cadastro de determinado pedido.

Figura 4.9 - Refinamento em PFS das atividades [Novo Pedido] e [Funções de Registro de Pedido]

Figura 4.10 - Refinamento em PFS das atividades [Seleciona Critério] e [Funções de Registro de Critério para Pedido]

Os modelos funcionais em RdP das atividades [Novo Pedido] e [Funções de Registro de Pedido] são apresentados na Figura 4.11. A descrição dos elementos do modelo em RdP (*lugares* e *transições*), que representam as operações se encontram na Tabela 4.9 e Tabela 4.10 .

Considera-se, no modelo em RdP da Figura 4.11, que existe fusão de transição entre as transições T1a e T1b, e entre T5a e T2b.

Figura 4.11 - Modelo em RdP das atividades [Novo Pedido] e [Funções de Registro de Pedido]

Tabela 4.9 – Descrição dos elementos em RdP das atividades [Novo Pedido]

<u>Atividade</u> [Novo Pedido]	
Lugar	Descrição
L1a	Aguarda uma mensagem de requisição para cadastro de Novo Pedido
L2a	Mensagem de requisição de cadastro foi recebida
L3a	Preparação para execução de cadastro de novo pedido
L4a	Cadastro do novo pedido realizado
L5a	Preparação da mensagem de requisição para seleção de SPs
L6a	Mensagem de requisição de seleção de SPs enviada
L7a	Aguarda mensagem com lista de SPs selecionados para o pedido
L8a	Mensagem com lista de SPs selecionados para o pedido foi recebida
L9a	Preparação da mensagem de cadastro de pedido realizada
L10a	Mensagem de cadastro de novo pedido realizada
Transição	Descrição
T1a	Transição requisitada para cadastro de novo pedido
T2a	Transição interna de cadastro de novo pedido
T3a	Transição requisitante para seleção de lista de SPs
T4a	Transição de resposta requisitante com lista de SPs
T5a	Transição de resposta de cadastro de novo pedido

Tabela 4.10 - Descrição dos elementos em RdP da atividade
[*Funções de Registro de Pedido*]

<u>Atividade</u> [<i>Funções de Registro de Pedido</i>]	
Lugar	Descrição
L1b	Preparação da mensagem para solicitar cadastro de um novo pedido
L2b	Mensagem de solicitação de cadastro de novo pedido enviada
L3b	Aguarda mensagem com cadastro do pedido (numero do pedido)
L4b	Mensagem com cadastro do pedido foi recebida
Transição	Descrição
T1b	Transição requisitante para cadastro de pedido
T2b	Transição de resposta requisitante de cadastro de pedido

O modelo funcional em RdP das atividades [*Seleciona Critério*] e [*Funções de Registro de Critério para Pedido*] são apresentados na Figura 4.12. A descrição dos elementos do modelo em RdP (lugares e transições), que representam as operações se encontram na Tabela 4.11 e Tabela 4.12.

Considera-se que no modelo em RdP da Figura 4.12 existe fusão de transição entre as transições T1a e T1b, e entre T3a e T2b.

Figura 4.12 – Modelo em RdP das atividades [Seleciona Critério] e [Funções de Registro de Critério para Pedido]

Tabela 4.11 – Descrição dos elementos do modelo em RdP das atividades [Seleciona Critério] e [Funções de Registro de Pedido]

<u>Atividade</u> [Seleciona Critério]	
Lugar	Descrição
L1a	Aguarda uma mensagem de requisição para seleção de critério em Novo Pedido
L2a	Mensagem de requisição de seleção de critério para novo Pedido foi recebida

Lugar	Descrição
L3a	Preparação para execução de seleção de critério para novo pedido
L4a	Cadastro de critério do novo pedido realizado
L5a	Preparação da mensagem de cadastro de critério do novo pedido realizada
L6a	Mensagem de cadastro de critério do novo pedido realizada
Transição	Descrição
T1a	Transição requisitada para seleção de critério do novo pedido
T2a	Transição interna de seleção de critério de novo pedido
T3a	Transição de resposta de cadastro de critério para novo pedido

Tabela 4.12 - Descrição dos elementos do modelo em RdP da atividade [Funções de Registro de Critério para Pedido]

<u>Atividade</u> [Funções de Registro de Critério para Pedido]	
Lugar	Descrição
L1b	Preparação da mensagem para solicitar seleção de critério para um novo pedido
L2b	Mensagem de solicitação para seleção de critério de novo pedido enviada
L3b	Aguarda mensagem com critério de novo pedido selecionado
L4b	Mensagem de notificação do critério selecionado para o novo pedido foi recebida
Transição	Descrição
T1b	Transição requisitante para seleção de critério para o novo pedido
T2b	Transição de resposta requisitante de notificação de seleção de critério de novo pedido

4.2.2. Atividade [*Consulta Pedidos*]

O refinamento da atividade [*Consulta Pedidos*] está na Figura 4.13. Esta atividade inicia-se pela atividade [*Recebe Requisição de Consulta*], em seguida, [*Executa Busca de Pedidos*] e, por fim, [*Envia Lista de Pedidos*]. O ator “Observador de Pedidos” utiliza a atividade [*Consulta Pedidos*] para verificar quais os pedidos que ainda não estão finalizados e não foram atualizados por um período de tempo maior que o definido. Com esta consulta, ele identifica quais os pedidos (por meio dos números de pedidos) devem ser verificados por meio da atividade [*Disponibilidade de SPs*] descrita em seguida.

Figura 4.13 - Refinamento em PFS da atividade [*Consulta Pedidos*]

O modelo funcional em RdP da atividade [*Consulta Pedidos*] é apresentado na Figura 4.14. A descrição dos elementos do modelo em RdP (*lugares* e *transições*), que representam as operações se encontram na Tabela 4.13.

Figura 4.14 – Modelo funcional em RdP da atividade [Consulta Pedidos]

Tabela 4.13 – Descrição dos elementos do modelo em RdP da atividade [Consulta Pedidos]

<u>Atividade</u> [Consulta Pedidos]	
Lugar	Descrição
L1a	Aguarda uma mensagem de requisição para consulta de Pedidos
L2a	Mensagem de requisição para consulta de pedidos foi recebida
L3a	Preparação para execução de busca por pedidos
L4a	Busca por pedidos foi realizada
L5a	Preparação da mensagem com resultado da busca de pedidos
L6a	Mensagem com resultado da busca de pedidos foi enviada
Transição	Descrição
T1a	Transição requisitada para realização da busca de pedidos
T2a	Transição interna de busca de pedidos
T3a	Transição de resposta de resultado da busca pelos pedidos

4.2.3. Atividade [*Disponibilidade de SPs*]

A atividade [*Disponibilidade de SPs*] é responsável por coordenar o processo produtivo, entre os diferentes SPs, para o atendimento de determinado pedido. Conforme já explicado, a ideia é concentrar o *know-how* do processo produtivo nesta atividade, de maneira independente das demais atividades do sistema de orquestração de SPs proposto. Desta forma, quando um produto é modificado, por meio da mudança de seu processo produtivo, deve-se alterar apenas a atividade [*Disponibilidade de SPs*], o que facilita sua reconfiguração, auxiliando na redução do tempo necessário para o início de produção (*time-to-market*).

O refinamento da atividade [*Disponibilidade de SPs*] está na Figura 4.15. Depois de feita a requisição da atividade [*Disponibilidade de SPs*], é realizada a atividade [*Consulta Estado Pedido*], que retorna o estado atual de execução do pedido. Na Tabela 4.14, tem-se a descrição de cada estado possível para o determinado pedido. De acordo com o resultado da consulta, é iniciada a atividade [*Execução Etapa SP1*] ou as demais atividades [*Execução Etapa SPx*] que desencadeia a produção em cada um dos SPs necessários para o processo produtivo global do pedido. É o estado do pedido que determina qual é a *transição* a ser habilitada para execução da atividade em determinado SP. Por fim, é enviada, para o ator “*Observador de Pedidos*”, a mensagem de execução da atividade [*Disponibilidade de SPs*] por meio da atividade [*Notifica Execução de Disponibilidades*].

Figura 4.15 – Refinamento em PFS da atividade [Disponibilidade de SPs]

Tabela 4.14 - Tabela com estados dos pedidos

Situação Atual do Pedido		Situação Futura do Pedido	
Estado	Etapa	Estado	Etapa
Pendente	Não_Iniciado	Executando	Inicio_SP1
Executando	Inicio_SP1	Executando	Fim_SP1
Executando	Fim_SP1	Executando	Inicio_SP2
Executando	Inicio_SP2	Executando	Fim_SP2
...		...	
Executando	Fim_SPn	Finalizado	Pedido_Finalizado

O modelo funcional em RdP da atividade [Disponibilidade de SPs] é apresentado na Figura 4.16. A descrição dos elementos do modelo em RdP (*lugares* e *transições*) que representam as operações se encontram na Tabela 4.15.

Figura 4.16 – Modelo funcional em RdP para a atividade [Disponibilidade de SPs]

Tabela 4.15 – Descrição dos elementos em RdP para a atividade [Disponibilidade de SPs]

<u>Atividade</u> [Disponibilidade de SPs]	
Lugar	Descrição
L1a	Aguarda uma mensagem para início de execução de Disponibilidades de SP
L2a	Mensagem para início de execução de disponibilidades recebida
L3a	Preparação para envio da mensagem de requisição de consulta de estado do pedido
L4a	Mensagem com o estado do pedido recebida
L5a	Preparação da mensagem de requisição de execução de Etapa do SP1
L6a	Mensagem de requisição de execução de Etapa do SP1 enviada
L7a	Preparação da mensagem de requisição de execução de Etapa do SP2

Lugar	Descrição
L8a	Mensagem de requisição de execução de Etapa do SP2 enviada
Lna	Preparação da mensagem de requisição de execução de Etapa do SPn
L(n+1)a	Mensagem de requisição de execução de Etapa do SPn enviada
L9a	Prepara a mensagem de notificação de execução de disponibilidade
L10a	Mensagem de notificação de execução de disponibilidade enviada
Transição	Descrição
T1a	Transição requisitada para início de execução de Disponibilidade
T2a	Transição interna de consulta de estado de pedido
T3a	Transição interna para execução de etapa no SP1
T4a	Transição interna para execução de etapa no SP2
Tna	Transição interna para execução de etapa no SPn
T5a	Transição de resposta de execução de disponibilidade

4.2.4. Atividade [Execução Etapa SPx]

O refinamento em PFS da atividade [Execução Etapa SPx] pode observado na Figura 4.17. Cada uma das diferentes funcionalidades necessárias para o processo produtivo global do produto deve executar uma atividade [Execução Etapa SPx] correspondente, como descrito na atividade [Disponibilidade de SPs].

Após o [Início Execução Etapa SPx], é verificada a disponibilidade do SPx que foi selecionado previamente durante a execução da atividade [Novo Pedido] pela atividade [Seleção de SPs] do “Serviço de Integração e Descoberta”. Caso a resposta recebida pela atividade [Resposta Disponibilidade SPx] indique a indisponibilidade de determinado SP, é selecionado um novo SP, de acordo com o critério prioritário, por meio das atividades [Solicita SP de Funcionalidade Específica]

Tabela 4.16 – Descrição dos elementos em RdP para a atividade [Execução Etapa SPx]

<u>Atividade</u> [Execução Etapa SPx]	
Lugar	Descrição
L1a	Aguarda uma mensagem para início de execução de Etapa SPx
L2a	Mensagem para início de execução de Etapa SPx recebida
L3a	Preparação para envio da mensagem de requisição de disponibilidade de SPx
L4a	Mensagem de requisição de disponibilidade de SPx enviada
L5a	Aguarda a mensagem de resposta de disponibilidade de SPx
L6a	Mensagem com resposta de disponibilidade de SPx recebida
L7a	Prepara mensagem para solicitação de seleção de SP de funcionalidade específica
L8a	Mensagem de solicitação de seleção de SP foi enviada
L9a	Aguarda mensagem com resposta de SP selecionado
L10a	Mensagem com SP selecionado recebida
L11a	Prepara mensagem para requisição de pedido para SPx
L12a	Mensagem de Requisição de pedido para SPx enviada
L13a	Prepara mensagem de requisição para registro de histórico de pedido
L14a	Mensagem com registro de histórico do pedido enviada
L15a	Aguarda mensagem de registro de histórico de pedido concluído
L16a	Mensagem com registro de histórico de pedido concluído recebida
L17a	Prepara a mensagem de notificação de execução de etapa de SPx
L18a	Mensagem de notificação de execução de etapa de SPx enviada

Transição	Descrição
T1a	Transição requisitada para início de execução de etapa de SPx
T2a	Transição requisitante de consulta de disponibilidade do SPx
T3a	Transição de resposta requisitante com disponibilidade de SPx
T4a	Transição requisitante de seleção de SPx
T5a	Transição de resposta requisitante com SPx selecionado
T6a	Transição requisitante de pedido para SPx
T7a	Transição requisitante de registro de histórico de pedido
T8a	Transição de resposta requisitante de registro de histórico de pedido
T9a	Transição de resposta requisitada de execução de etapa de SPx

4.2.5. Atividade [Resposta Serviço SPx]

Cada funcionalidade distinta para o processo produtivo global de determinado produto deve estar associado a uma atividade [Resposta Serviço SPx] específica para o atendimento a um pedido de produção. O refinamento em PFS da atividade [Resposta Serviço SPx] está na Figura 4.19. Após receber a notificação de entrega de pedido por meio da atividade [Recebe Notificação de Entrega de Pedido], é iniciada a atividade [Correlação com Pedido], em que é verificado se esta notificação corresponde a um pedido em execução, caso seja, é iniciada a atividade [Registro Histórico Pedido]. Dessa maneira, registra-se no histórico de pedido a etapa atual do pedido, neste caso, "Fim_SPx" conforme Tabela 4.14. Por fim, a atividade [Envia Resposta de Registro de Entrega], envia mensagem de resposta sobre a notificação.

Figura 4.19 - Refinamento em PFS da atividade [Resposta Serviço SPx]

O modelo funcional em RdP da atividade [Resposta Serviço SPx] é apresentado na Figura 4.20. A descrição dos elementos do modelo em RdP (*lugares e transição*), que representam as operações se encontram na Tabela 4.17.

Figura 4.20 - Modelo funcional em RdP da atividade [Resposta Serviço SPx]

Tabela 4.17 – Descrição dos elementos em RdP para a atividade [Resposta Serviço SPx]

<u>Atividade</u> [Resposta Serviço SPx]	
Lugar	Descrição
L1a	Aguarda mensagem requisitante de notificação de entrega de pedido do SPx
L2a	Mensagem requisitante para notificação de entrega de pedido recebida
L3a	Preparação de mensagem de verificação de correlação de pedido

Lugar	Descrição
L4a	Correlação de pedido verificada
L5a	Prepara mensagem de requisição para registro de histórico de pedido
L6a	Mensagem com registro de histórico do pedido enviada
L7a	Aguarda mensagem de registro de histórico de pedido concluído
L8a	Mensagem com registro de histórico de pedido concluído recebida
L9a	Prepara a mensagem de resposta de registro de resposta de SPx
L10a	Mensagem de resposta de registro de resposta de entrega de pedido de SPx enviada
Transição	Descrição
T1a	Transição requisitada para notificação de entrega de pedido de SPx
T2a	Transição interna de verificação de correlação de pedido
T3a	Transição requisitante de registro de histórico de pedido
T4a	Transição de resposta requisitante de registro de histórico de pedido
T5a	Transição de resposta requisitada de registro de entrega de pedido de SPx

4.2.6. **Atividade [Registra Histórico Pedido]**

O refinamento em PFS da atividade [Registro Histórico Pedido] está na Figura 4.21. Trata-se do modelo da atividade responsável por registrar o estado do pedido, e é requisitada pelas atividades [Execução Etapa SPx] e [Resposta Serviço SPx], Figura 4.17 e Figura 4.19 respectivamente.

Figura 4.21 - Refinamento em PFS da atividade [Registra Histórico Pedido]

O modelo funcional em RdP da atividade [Registra Histórico Pedido] é apresentado na Figura 4.21. A descrição dos elementos do modelo em RdP (*lugares e transições*) que representam as operações se encontram na Tabela 4.18.

Figura 4.22 - Modelo Funcional em RdP da atividade [Registra Histórico Pedido]

Tabela 4.18 - Descrição dos elementos em RdP da atividade [Registra Histórico Pedido]

<u>Atividade</u> [Registra Histórico Pedido]	
Lugar	Descrição
L1a	Aguarda mensagem requisitante para registro de Histórico de Pedido

Lugar	Descrição
L2a	Mensagem requisitante para registro de Histórico de Pedido
L3a	Preparação de mensagem de registro de Histórico de Pedido
L4a	Registro de Histórico de Pedido executado
L5a	Prepara a mensagem de resposta de confirmação de registro de histórico de pedido
L6a	Mensagem de confirmação de registro de histórico de pedido enviada
Transição	Descrição
T1a	Transição requisitada para registro de histórico de pedido
T2a	Transição interna de execução de registro de histórico de pedido
T3a	Transição de resposta requisitada de registro de histórico de pedido

---

4.3. SÍNTESE DO CAPÍTULO

Neste capítulo foram apresentados os modelos derivados da aplicação do procedimento PFS/RdP das atividades a serem realizadas dos principais componentes da arquitetura de controle para a orquestração de SPs dispersos, introduzida no Capítulo 3. Estes modelos representam a especificação para a implementação da arquitetura proposta.

5. EXEMPLO

Este capítulo apresenta um exemplo de implementação de alguns dos principais componentes da arquitetura de controle para a orquestração de SPs dispersos, introduzida no Capítulo 3. O objetivo aqui é ilustrar o procedimento adotado para testar, verificar e validar a arquitetura proposta e, nesse sentido, o estudo foi focado em certos componentes, funções e situações específicas.

5.1. SP DISPERSO CONSIDERADO

Como exemplo para implementação da arquitetura proposta para o sistema de controle de SP disperso utilizou-se o sistema flexível de montagem automatizado disponível e em operação no Laboratório de Sistemas de Automação (LSA) da EPUSP. Este sistema foi concebido para emular um processo produtivo composto por quatro SPs com funcionalidades distintas, cada um com certo grau de autonomia para execução de atividades produtivas e que quando devidamente coordenadas podem colaborar para a fabricação de produtos com diferentes características. Cada um destes SPs emula um SP disperso, e são denominados como SP de “Alimentação”, SP de “Inspeção”, SP de “Transporte” e SP de “Montagem” (Figura 5.1).

Figura 5.1 - Foto ilustrativa do sistema flexível e automatizado de "Alimentação", "Inspeção", "Transporte" e "Montagem" que emulam SPs dispersos

Neste exemplo, o SP “Alimentação” é responsável por fornecer peças “corpo” Figura 5.2, ao SP de “Inspeção”, que por sua vez executa o serviço de controle de qualidade e identificação das características físicas do produto fornecido pelo SP de “Alimentação” (peça “corpo”), de maneira que se garantam as especificações do produto requisitado pelo SP de “Montagem”. O SP de “Transporte” executa o serviço de transporte e movimentação dos *pallets* entre os outros SPs. Os *pallets* são onde as peças são colocadas. O SP de “Montagem”, por sua vez, executa o serviço de montagem dos produtos finais ou “peças montadas” que podem ser de três tipos: (i) “corpo” prata com “êmbolo” preto, “mola” e “tampa”; (ii) “corpo” rosa com “êmbolo” preto, “mola” e “tampa”; e (iii) “corpo” preto com “êmbolo” prata, “mola” e “tampa”. A quantidade de produtos finais é requisitada pelo o ator “*Cliente*”.

Figura 5.2 - Peças a serem montadas: corpo, pino, mola, tampa

No contexto de orquestração de SPs dispersos, com a arquitetura do sistema de controle proposta, o processo produtivo existente neste sistema de manufatura flexível é abstraído de maneira que cada uma das funcionalidades associadas ao SP de “Alimentação”, SP de “Inspeção”, SP de “Transporte” e SP de “Montagem”, possa ser realizada por mais de um SP, além do SP físico existente. Cada um destes SPs é exposto por um serviço implementado por WS, e estes WSs são integrados e interagem no sistema de controle desenvolvido. Por esse motivo, durante os testes e validação, utilizam-se programas computacionais, por meio de WSs, que atuam como os WSs de SPs com as funcionalidades necessárias para o processo produtivo estudado.

5.2. SERVIÇO DE INTEGRAÇÃO E DESCOBERTA

Conforme descrito nos capítulos anteriores, o “*Serviço de Integração e Descoberta*” é o serviço responsável por cadastrar novos SPs ao sistema de controle. Por meio da “*Interface de Integração*” o ator “*Integrador de SPs*” preenche o endereço do novo SP, o nome do serviço fornecido, o endereço de seu WSDL, o parâmetro de “*Binding*” (definindo qual é a versão do SOAP e o protocolo de transporte) e a sua funcionalidade padrão para o processo produtivo. O exemplo de implementação da interface, aplicada ao exemplo está ilustrado na Figura 5.3.

The screenshot shows a web browser window titled "Integrador_SPs - Windows Internet Explorer provided by IBM". The address bar shows "http://localhost:60300/Integrador_SPs.aspx". The page content includes a header with a diagram of a production system and the text "Integrador de Sistemas Produtivos". Below this is a "Cadastro" section with the following form elements:

- Nome do serviço:
- Endereço do Serviço:
- Endereço do WSDL:
- Funcionalidade: ☒ Alimentação ☐ Inspeção ☐ Transporte ☐ Montagem
- Protocolo: ☒ SOAP ☐ SOAP 1.2 ☐ HTTP POST
- Submit:

Figura 5.3 - Interface para o *Integrador de SPs*

Em seguida, os dados fornecidos pelo ator “*Integrador de SPs*” sobre o “novo SP” a ser cadastro são enviados para o método “*Comparador_WSDL*”, que por meio da análise do WSDL do WS do “novo SP”, avalia se este WS possui os métodos necessários para a determinada funcionalidade que ele possui e se estes métodos possuem nome, parâmetros de entrada e saída condizentes com o necessário para

a integração no sistema de controle. É importante frisar que é realizada uma análise sintática e supõe-se que o ator “Integrador de SPs” somente irá cadastrar SPs capazes de realizar a produção de acordo com a sua funcionalidade específica. Também é possível que um SP possua mais de uma funcionalidade diferente. Neste caso, o “Integrador de SPs” deve cadastrar o SP para cada funcionalidade diferente. Observa-se também que o “Integrador de Serviço”, por meio da “Interface de Integração”, seleciona o protocolo de comunicação e, assim, informa o formato dos dados (SOAP1.1, SOAP1.2, HTTP GET/POST) do WSDL deste “novo SP” . A seguir, na Figura 5.4, encontra-se um exemplo de implementação do método “Comparador_WSDL” elaborado em VB.NET¹⁹.

```
Public Function Comparador_WSDL(ByVal webServiceAddress As String, ByVal webServiceWSDLURL As
String, ByVal serviceName As String, ByVal protocolo As String, ByVal func As String) As String

    Dim client As System.Net.WebClient = New System.Net.WebClient()
    Dim client_referencia As System.Net.WebClient = New System.Net.WebClient()
    'Conecta-se com o WSDL do WS na web
    Dim stream As System.IO.Stream = client.OpenRead(webServiceWSDLURL)
    Dim stream_referencia As System.IO.Stream = client.OpenRead(webServiceWSDLURL)
    'Abre o stream do WS DE REFERENCIA de acordo com a funcionalidade selecionada
    If func = "Alimentacao" Then
        stream_referencia = client.OpenRead("http://localhost:56570/Alimentacao.asmx?WSDL")
    ElseIf func = "Inspecao" Then
        stream_referencia = client.OpenRead("http://localhost:56570/Inspecao.asmx?WSDL")
    ElseIf func = "Montagem" Then
        stream_referencia = client.OpenRead("http://localhost:56570/Montagem.asmx?WSDL")
    ElseIf func = "Transporte" Then
        stream_referencia = client.OpenRead("http://localhost:56570/Transporte.asmx?WSDL")
    End If
    Dim description_referencia As ServiceDescription =
ServiceDescription.Read(stream_referencia)
    Dim description As ServiceDescription = ServiceDescription.Read(stream)
    'Inicia um service description importer para o WS na WEB e o WS de referencia
    Dim importer As ServiceDescriptionImporter = New ServiceDescriptionImporter()
    Dim importer_referencia As ServiceDescriptionImporter = New ServiceDescriptionImporter()
    importer.ProtocolName = protocolo
    importer.AddServiceDescription(description, Nothing, Nothing)
    importer_referencia.AddServiceDescription(description_referencia, Nothing, Nothing)
    'Cria um proxy client.
    importer.Style = ServiceDescriptionImportStyle.Client
    'Inicia os valores das propriedades
    importer_referencia.Style = ServiceDescriptionImportStyle.Client
    importer.CodeGenerationOptions =
System.Xml.Serialization.CodeGenerationOptions.GenerateProperties
    importer_referencia.CodeGenerationOptions =
System.Xml.Serialization.CodeGenerationOptions.GenerateProperties
```

(continuação)

¹⁹ <http://www.microsoft.com/visualstudio/>

```

' Inicia a Code-DOM tree na qual o WS será importado
Dim nmspace As CodeNamespace = New CodeNamespace()
Dim unit1 As CodeCompileUnit = New CodeCompileUnit()
unit1.Namespaces.Add(nmspace)
Dim nmspace_referencia As CodeNamespace = New CodeNamespace()
Dim unit1_referencia As CodeCompileUnit = New CodeCompileUnit()
unit1_referencia.Namespaces.Add(nmspace_referencia)
' Cria o proxy code que usará o serviço
Dim warning As ServiceDescriptionImportWarnings = importer.Import(nmspace, unit1)
Dim warning_referencia As ServiceDescriptionImportWarnings =
importer_referencia.Import(nmspace_referencia, unit1_referencia)
If warning = 0 Then
    ' Gera o proxy code
    Dim provider1 As CodeDomProvider = CodeDomProvider.CreateProvider("VB")
    ' Compila os parametros da assembly proxy
    Dim assemblyReferences() As String
    assemblyReferences = New String() {"System.dll", "System.Web.Services.dll",
"System.Web.dll", "System.Xml.dll", "System.Data.dll"}
    Dim parms As CompilerParameters = New CompilerParameters(assemblyReferences)
    parms.GenerateInMemory = True
    Dim results As CompilerResults = provider1.CompileAssemblyFromDom(parms, unit1)
    Dim results_referencia As CompilerResults = provider1.CompileAssemblyFromDom(parms,
unit1_referencia)
    ' Verifica erros
    If results.Errors.Count > 0 Then
        Dim oops As CompilerError
        For Each oops In results.Errors
            System.Diagnostics.Debug.WriteLine("=====Compiler error=====")
            System.Diagnostics.Debug.WriteLine(oops.ErrorText)
        Next
        Throw New System.Exception("Erro durante a compilação.")
    End If
' Invoca o WS na web e o WS de referência
Dim wsvcClass As Object = results.CompiledAssembly.CreateInstance(servi ceName)
Dim wsvcClass_referencia As Object =
results_referencia.CompiledAssembly.CreateInstance(servi ceName)
Dim types As Type = wsvcClass.GetType
Dim types_referencia As Type = wsvcClass_referencia.GetType
Dim myArrayMethodInfo As MethodInfo() = types.GetMethods()
Dim myArrayMethodInfo_referencia As MethodInfo() = types_referencia.GetMethods()
Dim NomeMetodo As String = "Nome do Metodo"
Dim i As Integer
Dim j As Integer
Dim METODO_IGUAL As Boolean = False
' Realiza a comparação dos métodos do WS de referencia com o WS a ser inserido
For j = 0 To myArrayMethodInfo_referencia.Length - 1
    Dim myMethodInfo_referencia As MethodInfo =
CType(myArrayMethodInfo_referencia(j), MethodInfo)
    For i = 0 To myArrayMethodInfo.Length - 1
        Dim myMethodInfo As MethodInfo = CType(myArrayMethodInfo(i), MethodInfo)
        If myMethodInfo = myMethodInfo_referencia Then
            METODO_IGUAL = True
        End If
        NomeMetodo = myMethodInfo.Name
    Next i
    If METODO_IGUAL = False Then
        j = myArrayMethodInfo_referencia.Length - 1
    End If
Next j
' Se os métodos são iguais, realiza o cadastro do WS

```

(continuação)

```

' Se os métodos são iguais, realiza o cadastro do WS
If METODO_IGUAL = True Then
    Dim Cadastro_Ok As Boolean
    Cadastro_Ok = bdServicosSPs.CadastrarSP(func, servicoName, webServicoAddress,
webServicoWSDLUrl, protocolo)
    If Cadastro_Ok Then
        Return "SP Cadastrado com Sucesso"
        CadastrarEstadoSP(func, servicoName, webServicoAddress)
    Else
        Return "SP não pode ser cadastrado"
    End If
Else
    Return "SP não pode ser cadastrado"
End If

Else
    Return "SP não pode ser cadastrado"
End If

End Function

```

Figura 5.4 - Código em VB.NET com exemplo de implementação do método "Comparador_WSDL"

5.3. SERVIÇO DE COORDENAÇÃO

Conforme descrito nos capítulos anteriores, o “*Serviço de Coordenação*” é o serviço responsável por realizar a coordenação dos diferentes SPs, requisitados ao “*Serviço de Integração e Descoberta*”. A arquitetura proposta permite que, durante a execução de um pedido, se um SP selecionado se torna indisponível, é realizada a seleção de um novo SP, de acordo com um critério prioritário. Neste exemplo, o ator “*Cliente*” pode fazer um novo pedido por meio da “*Interface do Cliente*”, ilustrada na Figura 5.5. A “*Interface do Cliente*” permite ao “*Cliente*” selecionar qual o produto escolhido, dentre as 3 opções existentes e a quantidade desejada deste produto. O “*Cliente*” também deve selecionar qual é o critério prioritário a ser atendido para a escolha de um novo SP, para a situação em que o SP selecionado para a execução deste pedido, e se torna está indisponível. No caso do exemplo implementado, o critério prioritário pode ser o “custo” ou “qualidade”.

Para o desenvolvimento do “*Observador de Pedidos*”, criou-se uma aplicação que durante intervalos fixos de tempo realiza a chamada do método “*Consulta_Pedidos*”, que retorna quais os pedidos não finalizados e, que estão a

mais tempo sem ser atualizados. Com isto, o “*Observador de Pedidos*” realiza a chamada do método “*Disponibilidade_SP*” para cada pedido, desencadeando novamente o processo produtivo. Para isto, a tabela “*Pedido*” contém como *primary-key* o código do pedido (“codPedido” – tipo *Integer*), o campo com o estado do pedido (“estado” – tipo *string*), o campo com a etapa atual (“etapa” – tipo *string*) e, o campo com o horário da última atualização (“datahora” – tipo “*datetime*”).

Figura 5.5 - Exemplo de implementação da Interface do *Cliente*

5.4. DISCUSSÃO DOS RESULTADOS

Os testes foram conduzidos com SPs emulados por *software*, que respondem às solicitações dos “*Serviços de Coordenação*” e o “*Serviço de Integração e Descoberta*”. Esta estratégia foi adotada, pois o foco aqui está na avaliação funcional da arquitetura de controle proposta, implementada a partir dos modelos em

RdP aplicado a este caso de estudo, e que foram verificados e validados através de simulação²⁰, conforme desenvolvido no capítulo 4.

Para avaliar a inserção dinâmica de SPs com a sua integração na arquitetura, por meio do método “Comparador_WSDL” para diferentes WSs, de diferentes funcionalidades e tipos de implementação, locais e/ou remotos, realizou-se testes também com WSs disponíveis na *web*, principalmente com WSs listados no site “Directory of Public SOAP Web Services”²¹. Os testes confirmaram que é possível realizar a comparação e cadastro dos WSs, de diferentes servidores, e implementações, desde que o protocolo (“SOAP”, “SOAP1.2”) esteja especificado corretamente para que o seu WSDL possa ser interpretado.

Por fim, no caso do exemplo implementado, observa-se pela Figura 5.5 que a interface do “*Cliente*” desenvolvida permite que o critério prioritário possa ser escolhido entre “custo” ou “qualidade”, para que seja selecionado um novo SP em caso de indisponibilidade durante a execução do pedido, de acordo com este critério. Para esta implementação, não foi realizada nenhuma análise específica de qualidade de serviços dos WSs (QoS – *Quality of Services*), ou qualidade em termos de produção, por meio de indicadores de desempenho. Para os testes realizados, a qualidade dos serviços dos SPs foi calculada de acordo com o número de vezes que o serviço é utilizado, isto é, quanto maior o número de vezes, melhor é a qualidade dos serviços. Para a avaliação do funcionamento do sistema de controle, considera-se este critério adequado, porém numa situação prática, deve-se considerar uma forma mais efetiva para avaliar a “qualidade” dos serviços, uma vez que desta maneira, pode-se criar um “vício” em que sempre os mesmos SPs de determinada funcionalidade são selecionados, mesmo que opções melhores (de acordo com outros critérios de qualidade) possam existir.

---

²⁰ Ferramenta utilizada: http://www.r-drath.de/Home/Visual_Object_Net++.html

²¹ <http://www.service-repository.com/>

6. CONCLUSÕES

O presente trabalho abordou uma solução baseada na SOA, considerando as características especiais de SPs dispersos, isto é, sistemas geograficamente distribuídos que transformam fisicamente matérias-primas em produtos e cujas funcionalidades devem ser devidamente coordenadas para a produção desses produtos. Esta coordenação das funcionalidades é feita por meio do sistema de controle que realiza a orquestração dos SPs, expostos como serviço. A SOA foi escolhida como paradigma de estruturação do sistema devido à sua capacidade de homogeneizar um ambiente heterogêneo, em virtude de sua solução de encapsulamento do funcionamento de cada um de seus componentes, desde que de acordo com as premissas do SOA.

A solução adotada aqui de supervisão e controle por meio da técnica de orquestração permite, quando necessário, a seleção entre SPs com funcionalidades semelhantes e, a inserção dinâmica do SPs escolhido no sistema. Dessa maneira, assegura-se mais “agilidade e versatilidade” ao processo produtivo. Esta “agilidade e versatilidade” pode representar um menor tempo de resposta do sistema como um todo, uma vez que, caso algum SP fique indisponível, a orquestração selecionará outro SP de mesma funcionalidade, de acordo com o critério prioritário, sem a necessidade de uma intervenção para reprogramação dos serviços. A escolha pelo “*Cliente*”, deste critério prioritário, permite que quando ocorra uma indisponibilidade de um SP, outro SP possa ser selecionado evitando a parada do pedido para aguardar o retorno (i.e, disponibilidade) do SP, com isso o processo produtivo global é atendido de uma maneira mais eficiente.

Neste trabalho, em que cada entidade, isto é, o serviço do SP disperso, tem autonomia, se adequa a abordagem de composição de WS realizada pela avaliação sintática dos WSs, por meio de seus WSDLs. A inserção dinâmica de SPs é realizada por meio da comparação entre o WSDL do serviço do novo SP e o WSDL padrão para um serviço de um SP da funcionalidade desejada. Com isso, garante-se que o serviço do novo SP é capaz de se integrar e se comunicar com o sistema de controle para orquestração de SPs de maneira que as mensagens e métodos sejam

do tipo esperado e contenham a informação adequada. Por isso, julga-se que o ator “*Integrador de SPs*” cadastra somente SPs de sua confiança.

Outro ponto a ser destacado na arquitetura proposta é a existência do componente “*Mecanismo de Seleção*”. Dentro da arquitetura, este componente é independente e responsável por selecionar, para cada pedido realizado, todos os SPs de cada funcionalidade, e retornar os SPs selecionados ao “*Serviço de Coordenação*”, para que realize a coordenação do processo produtivo. Neste trabalho o foco não está nos mecanismos de *scheduling* ou de seleção de parceiros, porém foi previsto, dentro deste componente, a possibilidade de implementar essas soluções na arquitetura. Como este componente recebe a lista de SPs que são capazes de realizar cada funcionalidade e também recebe a solicitação para seleção de SPs para determinado pedido, é possível implementar algoritmos de seleção de parceiros otimizados (WU e SU, 2005), ou interagir com sistemas de *scheduling* (PESSOA *et al.*, 2012). Também, ao se especificar esta arquitetura, considerou-se o “*Agente Descobridor*” como o único componente responsável por realizar as interações de busca no “*Diretório ativo de serviços dos SPs*”, por isso, além de identificar a lista de SPs para cada funcionalidade ao componente “*Mecanismo de Seleção*”, ele também é responsável por selecionar, de acordo com o “critério prioritário” escolhido pelo “*Cliente*”, qual é o SP que deve ser utilizado em substituição ao SP que está por algum motivo indisponível. Para o caso de estudo implementado, por exemplo, os “critérios prioritários” foram “custo” e “qualidade”. Entretanto, o componente “*Agente Descobridor*” pode ser aprimorado para suportar critérios prioritários mais elaborados para seleção, como os indicadores de desempenho apresentados em (BALDO, RABELO e VALLEJOS, 2009). Desta maneira, tem-se a vantagem desta arquitetura ser concebida com estes dois componentes independentes, o “*Agente Descobridor*” e o “*Mecanismo de Seleção*”, pois ambos permitem o desenvolvimento e inserção de soluções para seleção de parceiros (“*Partner Selection*”), sem alterar a configuração dos demais componentes.

Por sua vez, o componente da arquitetura “*Coordenação de SPs*”, implementado mais especificamente pela atividade [Disponibilidade SPs], concentra o *know-how* do processo produtivo para determinado produto, isto é, para cada produto ele contém a sequência de funcionalidades que deve ser seguida para a execução do pedido. Por isso, ao se criar novos produtos ou modificar/personalizar

produtos existentes, deve-se alterar ou inserir neste componente apenas as novas capacidades necessárias. Isto permite que o tempo de reconfiguração seja menor, contribuindo para maior “versatilidade” do sistema.

Por fim, na definição e especificação da arquitetura do sistema de controle de SPs, o uso de uma abordagem de modelagem baseada em uma ferramenta formal, como é a RdP e o PFS, permitiu estruturar um procedimento para a construção dos modelos e também para a análise funcional das partes que compõem este sistema e a interação entre elas. A avaliação desta arquitetura e suas propriedades foi feita por meio da análise estrutural, em que se verificou a lógica do modelo e se as composições das partes e as relações entre elas são consistentes e completas ; e a análise comportamental, onde se examinou o comportamento dinâmico através de simulação dos modelos em RdP. Este procedimento para o desenvolvimento da arquitetura permite uma generalização dos resultados obtidos para outras aplicações e, baseado no porte e complexidade de SPs dispersos reais, entende-se que a abordagem deve ser explorada para a especificação da arquitetura de controle de SPs dispersos.

TRABALHOS FUTUROS

Dentre as várias possibilidades de trabalhos futuros que podem ser derivados, destacam-se alguns a partir das limitações existentes e não abordadas nesta solução desenvolvida:

- Estudo para abordar a dispersão entre os SPs, levando em consideração: a distância entre os SPs, o custo e tempo para transporte de insumos ou produtos finais;
- Análise da implementação da arquitetura considerando indicadores de desempenho e também a qualidade dos serviços (QoS) tanto do ponto de vista computacional como funcional do SP, para ampliar o filtro de seleção de SPs do “critério prioritário” em caso de indisponibilidade;
- Implementação e testes do “Mecanismo de Seleção” com o uso de algoritmos de seleção de parceiros e/ou integração com serviços de *scheduling*;

7. REFERÊNCIAS

ADAM, N. ; ATLURI, V.; HUANG, W.-K. Modeling and Analysis of Workflows using Petri Nets. **Journal of Intelligent Information Systems**, Boston, v. 10, n. 2, p. 131-158, March/April 1998.

BALDO, F.; RABELO, R. J.; VALLEJOS, R. V. A framework for selecting performance indicators for virtual organisation partners' search and selection. **International Journal of Production Research**, v. 47, n. 17, p. 4737-4755, September 2009. ISSN 0020-7543.

BASHIOUM, C. et al. OASIS. **Reference Model for Service Oriented Architecture 1.0**, 2 August 2006. Disponível em: <http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm>. Acesso em: 07 October 2011.

BINDIETRICH, A. J.; KIRN, S.; SUGUMARAN, V. A service-oriented architecture for mass customization - a shoe industry case study. **IEEE Transactions on Engineering Management**, v. 54, n.1, p. 190-204, February 2007.

BRAUER, W.; REISIG, W. Carl Adam Petri and Petri nets. **Informatik-Spektrum**, v. 54, n. 1, p. 369-374, 2006.

CALVEZ, J. P. **Embedded Real-Time Systems**. [S.l.]: John Wiley & Sons, 1993. ISBN: 0 471 93563 8.

CÂNDIDO, G. et al. Generic management services for DPWS-enabled devices. **35th Annual Conference of the IEEE Industrial Electronics Society (IECON 2009)**, Porto, Portugal, p. 3-5, 2009a.

CÂNDIDO, G. et al. SOA in reconfigurable supply chains: A research roadmap. **Engineering Applications of Artificial Intelligence**, v. 22, p. 939-949, 2009b. DOI:10.1016/j.engappai.2008.10.020.

CERAMI, E. **Web Services Essentials (O'Reilly XML)**. 1st Edition. ed. [S.l.]: O'Reilly Media, Inc., 2002. ISBN: 978-0-596-00224-4.

CHING-TORNG LIN, H. C.; PO-YOUNG, C. Agility index in the supply chain. **International Journal of Production Economics**, v. 100, n. 2, p. 285-299, 2006.

CHRISTENSEN, E. et al. Web Services Description Language (WSDL) 1.1. **World Wide Web Consortium (W3C) Note**, 2001. Disponível em: <<http://www.w3.org/TR/wsdl>>. Acesso em: 22 Fevereiro 2013.

CIANCETTA, F.; D'APICE, B.; GALLO, D. Plug-and-play smart sensor based on web service. **IEEE Journal of Sensor**, v. 7, n. 5, p.882-889, 2007.

COBIS. EC FP6 STREP CoBis Project, 2006. Disponível em: <<http://coindex.html>>. Acesso em: 08 Setembro 2011.

DE SOUZA, A. P.; RABELO, J. R. A model for dynamic services discovery over largely distributed providers based on QoS and business process contexts. **IEEE World Congress on Services**, Washington DC, USA, p.347-354, July 2011.

DELAMER, I. M.; LASTRA, J. L. M. Self-orchestration and choreography: towards architecture-agnostic manufacturing systems. **20th International Conference on Advanced Information Networking and Applications. IEEE Computer Society**, v.2, 2006.

D'MELLO, D. A.; ANANTHANARAYANA, V. S.; SALIAN, S. A review of dynamic web service composition techniques. **1st International Conference on Computer Science and Information Technology**, Bangalore, India, p.85-97. January 2011.

DOMNIC, S.; KARNOUSKOS, S. Web service enabled wireless sensors in SOA environments. **13th IEEE International Conference on Emerging Technologies and Factory Automation**, Hamburg, Germany, p. 952-958. September 2008.

ERL, T. **Service-Oriented Architecture: Concepts, Technology, and Design**. Prentice Hall/ Pearson PTR, p.760, 2006. ISBN: 0131858580.

GARCIA MELO, J. I. **Arquitetura com orientação a serviços para sistemas produtivos teleoperados e dispersos**. Tese de Doutorado. Escola Politécnica da USP. São Paulo, 129p., 2011.

GARCIA MELO, J. I. et al. Towards Modular and Coordinated Manufacturing Systems Oriented to Services. **Dyna.**, v. 77, n. 163, p. 201-210, September 2010.

GARCIA MELO, J. I.; MORALES, R. A. G.; MIYAGI, P. E. Supervisory system for hybrid productive system based on bayesian networks and OO-DPT nets. **Proc. of 13th IEEE International Conference on Emerging Technologies and Factory Automation**, Hamburg, Germany, p. 1108-1111. 2008.

GARRO, O.; MARTIN, P. Towards new architecture of machine tools. **International Journal of Production Research**, v. 31, n. 10, p. 2403-2414, 1993.

GOMES, L.; BARROS, J. Structuring and composability issues in petri nets modeling. **IEEE Transactions on Industrial Informatics**, v. 1, n. 2, p. 112-123, 2005.

GOTTSCHALK, K. Web Services architecture overview. **IBM developerWorks**, 2000. Disponível em: <<http://www.ibm.com/developerworks/library/w-ovr/>>. Acesso em: 28 September 2011.

GREFFEN, P. et al. Dynamic business network process management in instant virtual enterprises. **Computer in Industry**, v.60, n.2, p.86-103, February 2009. ISSN 0166-3615.

GROBA, C. et al. A service-oriented approach for increasing flexibility in manufacturing. **IEEE International Workshop on Factory Communication Systems**, Dresden, p. 415-422, 2008.

HASEGAWA, K. Modeling, control and deadlock avoidance of FMS. **XI Congresso Brasileiro de Automática**, São Paulo, p. 37-51, 1996.

HASEGAWA, K.; MIYAGI, P. E.; TAKAHASHI, K. **Mark FlowGraph (MFG) para modelamento e controle de sistemas a eventos discretos**. Monografias em Automação e Inteligência Artificial. São Paulo, p. 1-10. 1989.

HASEGAWA, K.; MIYAGI, P. E.; TAKAHASHI, K. Mark FlowGraph (MFG) para modelamento e controle de sistemas a eventos discretos. **Monografias em Automação e Inteligência Artificial**, São Paulo, p.1-10, 1989.

HERRON, C.; HICKS, C. The transfer of selected lean manufacturing techniques from Japanese automotive manufacturing into general manufacturing (UK) through change agents. **Robotics and Computer-Integrated Manufacturing**, v. 24, n. 4, p. 524-531, August 2008. ISSN 0736-5845.

HOLMQVIST, M.; DURCHSLAG, S.; BROWN, J. S. Process integration and web services, a case of evolutionary development in a supply chain. **Scandinavian Journal of Information Systems**, v. 16, n. 1, p. 117-144, 2004.

IMC-AESOP Project. **IMC-AESOP Project**, 2011. Disponível em: <<http://www.imc-aesop.eu>>. Acesso em: 22 Fevereiro 2013.

INLIFE. EC FP6 STREP InLife Project, 2007. Disponível em: <<http://www.uninova.pt/~inlife/>>. Acesso em: 08 Setembro 2011.

JAMMES, F.; SMIT, H. Service-oriented paradigms in industrial automation. **IEEE Transactions on Industrial Informatics**, v.1, n.1, p.62-70, 2005.

JENSEN, K. **Colored Petri nets: Basic Concepts, Analysis Methods and Practical Use**. Berlin: Springer Verlag, 1992.

JUNQUEIRA, F.; VILLANI, E.; MIYAGI, P. E. A platform for distributed modeling and simulation of productive systems based on petri nets and object-oriented paradigm. **10th IEEE International Conference on Emerging Technologies and Factory Automation**, Catania, v.1, n.1, p.907-914, 2005.

KANESHIRO, P. J. et al. Modeling of collision resolution algorithm in Lonworks networks. **Proc. of ASME International Mechanical Engineering Congress and Exposition**, Seattle, WA, 2007.

KIEPUSZEWSKI, B.; HOFSTEDE, A.H.M.; VAN DER AALST, W. M. P. Fundamentals of control flow in workflows, **Proc. of Workflows**. p.149-209, 2003.

KIM, W. et al. WSCPC: An architecture using semantic web services for collaboration product commerce. **Journal of Computers in Industry**, v.57, p. 787-796, 2006.

KIRKHAN, T. et al. SOA middleware and automation: Services, applications and architectures. **Proc. of 6th IEEE International Conference on Industrial Informatics**, Daejeon, Korea, p. 1419-1424, 2008.

KOMODA, N. Service oriented architecture (SOA) in industrial systems. **Proc. IEEE International Conference on Industrial Informatics**, Singapore, p. 1-5. 2006.

KUMAR, R. S. et al. Virtualization for manufacturing web services: a WS-RF approach. **International Journal of Information Technology**, v. 11, n. 3, p. 40-51, 2005.

LEE, J.; ZHOU, M.; HSU, P. An application of Petri nets to supervisory control for human-computer interactive systems. **IEEE Trans. on Industrial Electronics**, v. 52, n. 5, p.1220-1226, 2005.

LEITÃO, P. Agent-based distributed manufacturing control: A state-of-art survey. **Engineering Applications of Artificial Intelligence**, Bragança. Portugal, v. 22, p. 979-991, 2009. ISSN doi:10.1016/j.engappai.2008.09.005.

LI-FENG, W.; ZHOU, B. Service-oriented software architecture for flexible manufacturing control system. **IEEE International Conference on Automation Science and Engineering**, Shangai, p.425-430, 2007.

LIU, S. et al. Design and implementation of a service-oriented manufacturing grid system. **Proc. of Congress of Advanced Web and Network Technologies and Applications**, v.3842, p. 643-647, 2006.

LIU, W.; MIYAGI, P. E.; SCHREEK, G. MFG/PFS methodology in manufacturing industries. **Studies in Informatics and Control**, v. 3, n. 2-3, p. 195-200, 1994.

MADHUSUDAN, T. An experience report on developing an automated web services platform. **Journal of Information Systems and E-business Management**, v. 3, n. 3, p. 243-263, 2005.

MARTINS, R. S. D. M. **Composição dinâmica de Web Services**. Dissertação de Mestrado. Universidade do Vale do Rio do Sinos. São Leopoldo, 77p., 2007.

MAYERL, C.; VOGEL, T.; ABECK, S. SOA-based integration of IT service management applications. **IEEE International Conference on Web Services**, Orlando, FL, p.785-787, 2005.

MEDINA, I. et al. Proposal of a Methodology for Implementing a Service-Oriented Architecture in Distributed Manufacturing Systems. **3rd Manufacturing Engineering Society International Conference**, v.1181, p.622-632, 2009.

MEHRABI, M. G.; ULSOY, A. G.; KOREN, Y. Reconfigurable manufacturing systems: Key to future manufacturing. **Journal of Intelligent Manufacturing**, Ann Arbor, v. 11, n. 4, p. 409-419, 2000. ISSN doi: 10.1023/A:1008930403506.

MELO, J. I. G. et al. A systematical approach to expose manufacturing system as a service. **9th IEEE/IAS International Conference on Industry Applications**, Sao Paulo, p.1-6, November 2010

MENDES, J. et al. Engineering of service-oriented automation systems: a survey. **Proc. of Conference on Innovative Production Machines and Virtual International Systems**, p.33-38, 2007.

MENDES, J. M. et al. Service-Oriented Control Architecture for Reconfigurable Production Systems. **Proc. of IEEE International Conference on Industrial Informatics**, Daejeon, Korea, p.744-749, 2008.

MENG, L.; YEXIN, X. SOMG: a service oriented manufacturing grid system. **5th International Conference on Computer and Information Technology**, Shangay, China, p.1080-1084, September 2005.

MINOR, J. **Bridging OPC UA and DPWS for Industrial SOA**. Tese de Mestrado. Tampere University of Technology. 80p. 2012.

MIYAGI, P. E. **Controle Programável: Fundamentos do Controle de Sistemas a Eventos Discretos**. 1a Edição. ed. São Paulo: Edgar Blucher, 194p., 1996. ISBN: 85-212-0079-X.

MORAES, C. C. D.; CASTRUCCI, P. D. L. **Engenharia de Automação Industrial**. 2a. ed. ed. Rio de Janeiro: LTC, 2007. ISBN ISBN 978-85-216-1532-3.

MORALES, R. G.; MIYAGI, P. E.; GARCIA MELO, J. I. Diagnosis and treatment of faults in production systems based on Bayesian networks and Petri net. **Proc. of IEEE Conference on Automation Science and Engineering**, Scotsdale, AZ, 2007.

MOREIRA, L. et al. SOCRADES: A web service based shop floor Integration Infrastructure. **Proc. of Internet of Things 2008 Conference**, Zurich, Switzerland, 2008.

NAGORNY, K.; COLOMBO, A. W.; SCHIMIDTMANN, U. A service- and multi-agent-oriented manufacturing automation architecture: An IEC 62264 level 2 compliant implementation. **Computers in Industry**, v. v.63, n. Issue 8, p. 813-823, October 2012.

NASSAR, M. G. V. et al. Modeling and Analysing of the Material Entry Flow System in a Pickling Line Process using Petri Net. **ABCM Symposium Series in Mechatronics**, Brasilia, p. 444-453, 2008.

OASIS. OASIS. **Web Services Dynamic Discovery (WS-Discovery) Version 1.1**, 2009. Disponível em: <<http://docs.oasis-open.org/ws-dd/discovery/1.1/wsdd-discovery-1.1-spec.html>>. Acesso em: 08 Setembro 2011.

OPC. OPC Unified Architecture (OPC-UA) Specifications, 2008. Disponível em: <<http://www.opcfoundation.org/UA>>. Acesso em: 08 Setembro 2011.

OYANG, C. et al. Formal semantics and analysis of control flow in WS-BPEL. **Science of Computer Programming**, v. 67, n. 2, p. 162-198, 2007.

PESSOA, M. A. O. et al. Virtual Enterprise Planning System using time windows and capacity constraint concepts. **IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society**, Montreal, QC, p.2851-2856. October 2012.

PHAITHOONBUATHONG, P. et al. Web services based automation for the control and monitoring of production systems. **International Journal of Integrated Manufacturing**, v. 23, n. 2, p. 126-145, 2010.

PINTO, C. C. S. et al. Composição dinamica de Web-Services. **XXII Simpósio Brasileiro de Engenharia de Software**, Campinas-SP, p.220-235, 2008.

PUTTONEN, J. et al. A application of BPEL for service orchestration in an industrial environment. **IEEE International Conference on Emerging Technologies and Factory Automation**, p.530-537, 2008.

RAHANI, A.; AL-ASHRAF, M. Production flow analysis through value stream mapping: a lean manufacturing process case study. **Procedia Engineering**, Kuching, Sarawak, Malaysia, v. 41, p. 1727-1734, September 2012. ISSN 1877-7058.

REISIG, W. **A Primer in Petri Design**. Berlin: Springer-Verlag, 1992.

RIASCOS, L. A.; MIYAGI, P. E. Supervisor system for detection and treatment of failures in balanced automation systems using Petri nets. **Proc. of IEEE International Conference on Systems, Man and Cybernetics**. v.4, p. 2528-2533, Tucson, USA. 2001.

RI-MACS. **IST NMP-1 RI-MACS Project website**, 2007. Disponível em: <<http://www.rimacs.org>>. Acesso em: 08 Setembro 2011.

ROZENBERG, G.; ENGELFRIET, J. Elementary net systems. **Computer Science**, v. 1494, p. 12-121, 1996.

SANTOS FILHO, D. J. **Sistemas antropocêntricos de produção baseado em redes de Petri**. Tese de Doutorado, Escola Politécnica da Universidade de São Paulo. São Paulo, 287 p., 1998.

SHAN, J. et al. Research and Implementation of Code Generator for Information System based on SOA. **International Conference on Scalable Computing and Communication; 8th International Conference on Embedded Computing**, p.143-147, 2009.

SIRENA. ITEA 02014 SIRENA Project , 2006. Disponível em: <<http://www.sirena-itea.org>>. Acesso em: 08 Setembro 2011.

SOCRADES. EU FP6 IST-5-034116 SOCRADES Project, 2008. Disponível em: <<http://www.socrades.eu>>. Acesso em: 08 Setembro 2011.

SODA. ITEA 05022 SODA Project, 2007. Disponível em: <<http://www.soda-itea.org>>. Acesso em: 08 Setembro 2011.

SUA, Y.; GUOA, R.; CHANGB, S. Virtual fab: an enabling framework and dynamic manufacturing service provision mechanism. **Journal of Information & Management**, v. 42, p. 329-348, 2005.

SUN, L.; DONG, H.; ASHRAF, J. Survey of service description languages and their issues in cloud computing. **8th International Conference on Semantics, knowledge and grids**, Beijing, p.128-135. October, 2012.

TANENBAUM, A. S.; STEEN, M. V. **Distributed Systems: Principles and Paradigms**. 2nd Edition. ed. Amsterdam: Pearson Prentice Hall, 704 p., 2007. ISBN: 0-13-239227-5.

TRAVERSO, P.; PISTORE, M. Automated composition of semantic web services into executable processes. **The Semantic Web - ISWC 2004**, p.380-394, 2004.

VAN DER AALST, W. M. P. The application of Petri nets to workflow management. **Journal of Circuits, Systems and Computers**, v. 8, n. 1, p. 21-66, 1998.

VAN DER AALST, W. M. P. Pi calculus versus Petri Nets: Let us eat humble pie rather than further inflate the pi hype, 2003. Disponivel em: <<http://is.tm.tue.nl/research/patterns/download/pihype.pdf/>>. Acesso em: 08 September 2011.

VILLANI, E.; MIYAGI, P. E.; VALLETE, R. **Modelling and Analysis of Hybrid Supervisory Systems: A Petri Net Approach**. 1st Edition. ed. London: Springer, 2007.

WANG, H.; HU, Q. Research and application of an intergration platform for e-commerce system based on SOA. **International Conference on Management of e-Commerce and e-Government**, ICMECG. Nanchang, China, p.424-427, 2009.

WU , N.; SU, P. Selection of partners in virtual enterprise paradigm. **Robotics and Computer-Integrated Manufacturing**, v. 21, p. 119-131, 2005. ISSN 0736-5845.

YAN, Y.; LIANG, Y.; DU, X. Controlling remote instruments using Web services for online experiment systems. **Proc. of IEEE International Conference on Web Services**, 1, p.724-732, 2005.

ZEEB, E. et al. Service-Oriented Architectures for Embedded Systems Using Device Profile for Web Services. **Proc. of the 21st International Conference on Advances Information Networking and Applications**, p.956-963, 2007.

ZHANG, R.; ARPINAR, B.; ALEMAN-MEZA, B. Automatic composition of semantic web services. **1st Internation Conference on Web Services**, p.38-41, 2003.

ZHOU, M.; VENKATESH, K. **Modeling, simulation and control of flexible manufacturing systems: a Petri net approach**. World Scientific Publishing Co. pte. Ltd, 1999.

ZHOW, M. C.; LI, Z. W. Control of Elementary and Dependent Siphons in Petri Nets and Their Application. **IEEE Transactions on Systems, Man, and Cybernetics**, v. 38, n. 1, Janeiro 2008. ISSN 1083-4427.

ZHU, W.; MELO, W. Refactoring J2EE application for JBI-based ESB: A case of study. **IEEE International Enterprise Distributed Object Computing Conference**, Auckland, Nova Zelândia, p.213-217, 2009.

ZURAWSKI, R.; ZHOU, M. Petri nets and industrial applications: a tutorial. **IEEE Transactions on Industrial Electronics**, v. 41, n. 6, p.567-583, 1994.