4
Estudos de Caso

Os estudos de caso foram desenvolvidos com o intuito de ganhar subsídios e procurar entender os principais problemas recorrentes em aplicações baseadas em serviços.

4.1.
Busca Semântica

Aplicações de busca estão entre as mais populares da Web, sendo utilizadas regularmente pela imensa maioria dos usuários. Entretanto, grande parte dos usuários de máquinas de busca não fica satisfeita com os resultados apresentados por estas aplicações.

A principal razão está na maneira como a busca é realizada, utilizando um conjunto de termos que é então procurado em um conjunto de documentos (tipicamente, um pedaço indexado da Web). Este tipo de busca, denominado *Full-text search* (referência FTS), oferece diversas limitações por desconsiderar os diversos sentidos e contextos em que cada termo pode aparecer. Por esta razão, uma busca por uma determinada palavra trará todos os documentos que apresentam o termo, independente do sentido daquele termo no documento.

Um exemplo deste tipo de problema pode ser visto quando um usuário, interessado em páginas sobre culinária, digita o termo *rio* em alguma máquina de busca. Como resultado, são apresentadas páginas sobre a cidade do rio de janeiro misturadas a páginas sobre o rio Amazonas, o zagueiro da seleção inglesa Rio Ferdinand e propriedades de um rio (termo geofráfico), entre outros.

Tomando este problema como ponto de partida, este estudo de caso tinha como objetivos (i) determinar como ontologias poderiam permitir que a aplicação realizasse inferências sobre os termos procurados, obtendo o sentido pelo qual o usuário procura e (ii) como utilizar um conjunto de serviços que abstraíssem fontes de informações diferentes, permitindo que a busca fosse direcionada a partir do sentido selecionado.

A ontologia básica desta aplicação contém apenas três classes: *conceito, serviço de busca* e *termo*. Uma instância de termo pode estar associada a várias instâncias de conceito diferentes, representando os diferentes contextos em que uma busca pode ser feita com aquele termo. A classe *serviço de busca* também está associada a conceito e determina que serviços de busca são aplicáveis para buscas relativas a um conceito. A figura abaixo apresenta a ontologia básica desta aplicação.

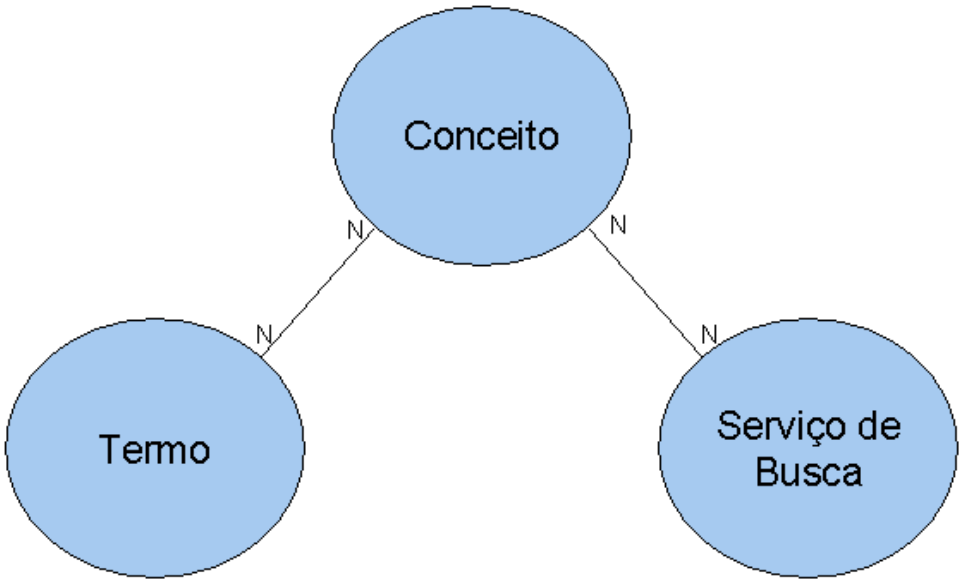

Figura 6 – Ontologia básica da aplicação de busca semântica

De forma geral, o uso de ontologias permite a formação de uma base de conhecimento sobre as palavras, associando-as a seus diferentes sentidos presentes em diferentes contextos. No entanto, a tentativa de usar a ontologia como forma de auxiliar a composição dinâmica de serviços permite que a seleção das fontes de dados seja mais adequada a cada contexto.

Por exemplo, se um dado usuário procura por um termo que é título de um livro, pode ser interessante procurar esse livro em uma loja de livros, ao invés de numa máquina de busca qualquer. Além disso, é possível enriquecer a consulta feita aos serviços com informações relativas ao conceito sugerido, o que pode aumentar a precisão das respostas.

Apesar de interessante, a composição de serviços de informação diferentes traz um novo desafio, a necessidade de integração de diferentes representações de conhecimento. Uma vez que usuários procuram por qualquer tipo de informação, são necessárias diversas ontologias que representem diferentes domínios. Por esta razão, elas precisam ser compostas de alguma forma.

Outro problema recorrente é a forma de invocar o serviço. Como cada um deles utiliza uma maneira diferente de organizar a informação, a chamada a cada um deles precisa ser estruturada de maneira diferente.

4.1.1. Arquitetura da aplicação

A arquitetura da aplicação de busca procurava utilizar a ontologia básica do sistema para integrar diferentes ontologias de termos e conceitos em um único lugar. Utilizando diferentes serviços, é possível formar dinamicamente uma ontologia a partir de diferentes serviços de ontologias que indiquem os diferentes conceitos que um termo pode assumir.

A aplicação possui uma taxonomia interna que associa cada termo aos diferentes conceitos das diferentes ontologias, permitindo que todos sejam integrados em uma única busca. A figura a seguir apresenta a arquitetura da aplicação *SemanticSearch*:

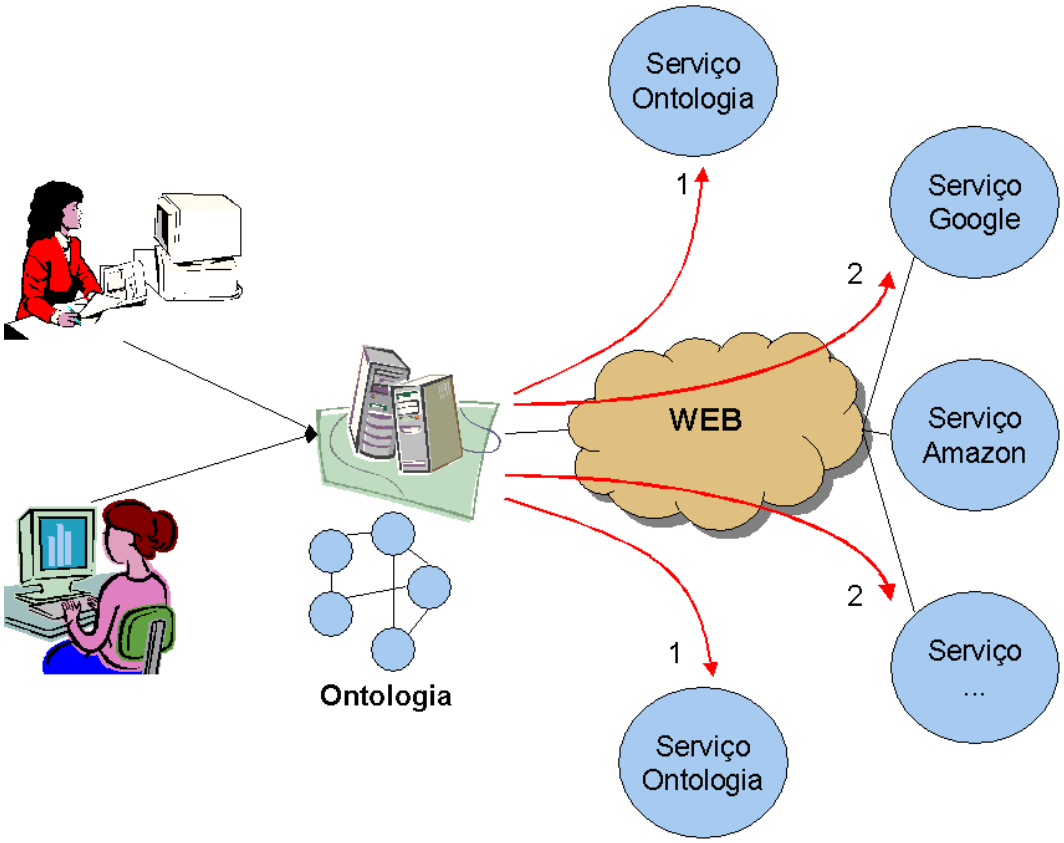

Figura 7 - Arquitetura da aplicação de busca (1) a aplicação de busca consulta serviços de ontologia para descobrir os conceitos aos quais o termo procurado está associado. Em um segundo momento (2), os serviços de informação associados são consultados

para obter referências para páginas web que possuam conteúdo relacionado ao termo procurado.

4.1.2. Conclusões

O desenvolvimento da aplicação de busca semântica levantou uma série de questões com relação à forma de composição dos dados de diferentes fontes. Poucos serviços adotam linguagens de descrição de ontologias para descrever o conteúdo, o que leva a uma integração manual dos dados.

Por esta razão, a aplicação de busca semântica necessita de uma representação canônica capaz de integrar diferentes ontologias de palavras em um único ponto de busca. No entanto, uma vez mapeados os conceitos, era possível capturar atualizações nas ontologias automaticamente. Por exemplo, se um novo termo, ligado ao conceito filme, é adicionado à ontologia A, seu respectivo conceito é automaticamente incluído na lista de opções quando uma busca pelo termo é realizada.

Por outro lado, é importante ressaltar a ausência de mecanismo único de representação do conhecimento compartilhado por todas as ontologias de palavras. Isto implica na criação manual do mapeamento entre os conceitos da aplicação e os presentes nas ontologias, um processo que é sujeito a erros.

O grande problema é a falta de informações semânticas nos serviços de informação. Boa parte deles permite a consulta apenas por palavras-chave, sem a possibilidade de uso das ontologias de palavras, o que aumentaria a precisão das respostas. Uma solução para este problema seria o uso de linguagens de descrição de ontologias como RDF/RDFSchema e DAML + OIL para a requisição e recebimento das informações provenientes dos serviços de informações.

Entretanto, além da inexistência de serviços capazes de prover tais informações, a infra-estrutura de software existente para publicação e acesso aos serviços não oferece suporte a dados descritos nestas linguagens, impedindo sua validação, aumentando o custo de processamento tanto nos clientes como nos provedores de serviços.

4.2. Estudo de Caso: Calendário

A marcação de compromissos e reuniões ainda é (surpreendentemente) uma tarefa difícil. A integração de diferentes sistemas de calendário ocorre de maneira fraca e bastante limitada, sendo é virtualmente impossível agendar compromissos com convidados que utilizam diferentes sistemas de calendário, exceto para os casos mais triviais.

Imaginando uma situação onde João, que trabalha numa companhia A e utiliza o Microsoft Outlook como seu cliente para e-mail, pretende agendar um compromisso com Maria e José de uma dada companhia B. São necessárias duas horas para a realização do compromisso que precisa ocorrer na semana de 24 de maio de 2003. Infelizmente, a empresa B adota o IBM Lotus Notes como serviço de calendário. Além de integrar as bases de dados dos dois sistemas, é necessário trocar diversas mensagens entre as partes de forma a encontrar o melhor espaço para agendar o compromisso. Para tal, é necessária a existência de algum serviço que proveja um conjunto de informações de calendário para os diferentes sistemas. Isto torna necessária a criação de mecanismos de coordenação que realizem essa tarefa com o mínimo de interação humana.

Para garantir uma forma simples de gerenciamento de calendário, um dos principais requisitos é a integração de diferentes sistemas de calendário, como o Microsoft Outlook, IBM Lotus Notes e Macintosh iCal.

O primeiro passo para realizar esta integração é a definição de um serviço de calendário padrão capaz de agendar e listar compromissos. Após esta definição, torna-se necessário expor as interfaces disponibilizadas por cada sistema de calendário como o serviço de calendário padrão. Cada serviço deve então ser disponibilizado através da utilização de um provedor de serviços. Este provedor de serviços deve então receber requisições no formato de algum protocolo de acesso (por exemplo, SOAP), utilizando algum protocolo de transporte (por exemplo, SMTP ou HTTP).

Quando um usuário precisa agendar compromissos com uma série de usuários, ele deve invocar o serviço coordenador de calendários para iniciar um compromisso multi-usuário. O coordenador é responsável por compor diferentes implementações distribuídas de serviços de calendário.

Para realizar as tarefas de coordenação, foi necessário desenvolver um serviço de gerenciamento de calendário em grupo. Um algoritmo "majority consensus" (Thomas, 1979) foi utilizado para agendar compromissos em grupo e obter consenso em situações de conflito. O algoritmo de coordenação utilizado é independente dos serviços de calendário, podendo ser instanciado de outra forma em outros serviços de gerenciamento de grupos.

É razoável destacar que a construção de diferentes algoritmos de coordenação poderia ser acelerada com a existência de um framework de coordenação capaz de facilitar o tratamento das respostas recebidas durante o processo de agendamento. Um núcleo comum formado por estruturas básicas de coordenação (fork, join, etc) permitiria que desenvolvedores tivessem reduzido esforço de programação, uma vez que todo o controle da composição funcional dos diferentes serviços precisa ser codificado no serviço de coordenação.

Outra necessidade seria a abstração dos diferentes protocolos de transporte, permitindo que as mensagens fossem tratadas pela aplicação independente do protocolo utilizado.

Visando testar os serviços implementados, foi desenvolvido um cliente Web, permitindo que os usuários iniciassem processos de marcação de compromissos em grupo a partir de uma única aplicação. O cliente Web também permite que usuários possam checar suas agendas e agendar compromissos tanto em grupo quanto isoladamente. A figura abaixo apresenta algumas telas do cliente Web para o serviço de calendário.

Como forma de testar a integração entre diferentes sistemas de calendários, foram desenvolvidos serviços de calendário para o Microsoft Outlook e um serviço de calendário proprietário baseado em IBM DB2.

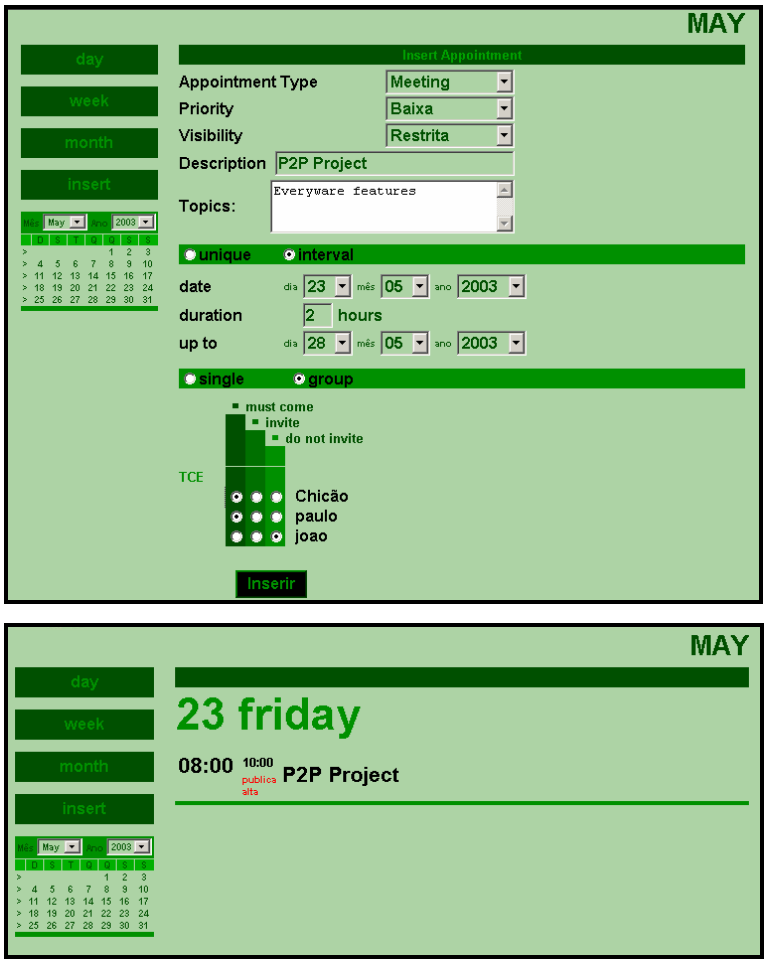

Figura 8 - O cliente Web do sistema de calendários baseado em serviços

Um dos problemas encontrados durante o desenvolvimento desta aplicação foi a indisponibilidade dos serviços durantes o processo de composição. Boa parte dos serviços de calendário está disponibilizada em PCs, telefones celulares ou PDAs, que podem ser desligados ou desconectados da rede de tempos em tempos.

Por esta razão, optou-se por criar uma arquitetura distribuída dos serviços de coordenação. Desta forma, é possível colocar servidores específicos para orquestração, garantindo que mensagens de resposta enviadas por serviços com baixa disponibilidade fossem processadas e que não se perdessem devido a eventuais falhas no coordenador. Para cada serviço de coordenação existe um conjunto de um ou mais serviços que recebem réplicas de todas as informações de coordenação. Isto permite que o processo de coordenação continue em caso de falha do coordenador. Neste caso, um processo de eleição de um novo coordenador é necessário. Esta eleição pode ser feita entre os serviços participantes da aplicação ou em catálogos de serviços (por exemplos, registros UDDI).

Outro problema recorrente durante o desenvolvimento da aplicação de calendário foi a representação heterogênea das informações fornecidas por cada sistema de calendário. Cada sistema tem uma forma própria de representar compromissos, o que implicava conversões e adaptações para cada serviço. Como exemplo, podemos citar serviços que descrevem compromissos através de uma data inicial e uma duração precisando ser adaptados para fornecer a data inicial e data final. Além disso, a ordem dos parâmetros na chamada dos serviços também pode afetar na sua execução. Por esta razão, cada cliente precisa conhecer a ordem e os tipos associados a cada parâmetro de entrada nos serviços, de forma a garantir que estes serão interpretados corretamente por cada serviço. Existem ferramentas que permitem compromissos com mais de um dia de duração, enquanto outra não. Neste caso, um compromisso precisa ser quebrado em uma série de compromissos de um dia.

Tentando resolver essa questão e buscando dar semântica aos serviços, foram desenvolvidos serviços baseados em ontologias de agendas para permitir que as informações trocadas por cada parte pudessem ser processadas corretamente. Foram utilizadas duas ontologias para a realização de testes com semântica nos serviços, a (Hybrid), baseada no padrão (iCal) da (IETF) e a (DAML Agenda). Um serviço de conversão foi implementado, de forma a garantir que serviços baseados na ontologia Hybrid pudessem interpretar ontologias DAML Agenda. A representação de um compromisso segundo as duas ontologias é apresentada nas duas figuras abaixo:

```
<rdf:RDF xmlns:ical="http://ilrt.org/discovery/2001/06/schemas/ical-full/hybrid.rdf#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

<ical:VEVENT rdf:about="http://www.inf.puc-rio.br/chicao/events/115">
   <ical:DTSTART>
      <ical:DATE-TIME>
         <rdf:value>20030525T090000</rdf:value>
      </ical:DATE-TIME>
  </ical:DTSTART>
  <ical:DTEND>
     <ical:DATE-TIME>
         <rdf:value>20030525T094500</rdf:value>
     </ical:DATE-TIME>
  </ical:DTEND>
  <ical:DESCRIPTION>Reunião Brainstorm - Projeto Serviços</ical:DESCRIPTION>
</ical:VEVENT>
</rdf:RDF>
```

Figura 9 – A representação de um compromisso utilizando a ontologia iCal definida em RDFSchema

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<rdf:RDF
  xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs ="http://www.w3.org/2000/01/rdf-schema#"
  xmlns="http://www.daml.org/2001/10/agenda/agenda-ont#"
>

   <Meeting rdf:ID="http://www.inf.puc-rio.br/chicao/events/115">
     <day>
       <Day>
         <start>2003-05-25T09:00:00</start>
         <items rdf:parseType="daml:collection">
            <Block>
              <theme>Reunião Brainstorm - Projeto Serviços</theme>
              <items rdf:parseType="daml:collection"/>
              <duration>PT45M</duration>
            </Block>
          </items>
       </Day>
     </day>
   </Meeting>

</rdf:RDF>
```

Figura 10 – Um compromisso representado na ontologia DAML Agenda

Os provedores de Web Services mais comuns (Apache Axis, Apache SOAP, WASP, Systinet, entre outros) não oferecem suporte a ontologias, impossibilitando seu uso com os serviços. Por esta razão, assim como no estudo de caso de busca, não foi possível verificar se uma ontologia passada a um serviço está em conformidade com o seu esquema associado, pois todos os provedores recebem a ontologia e a interpretam como uma cadeia de caracteres.

Para contornar este problema, foi desenvolvida uma extensão para o provedor Apache SOAP, que é capaz de receber e enviar ontologias em formato RDF/RDFSchema ou DAML para os serviços disponibilizados. Esta extensão permite que as ontologias sejam validadas pelo próprio provedor de serviços, garantindo a conformidade com os esquemas e com os axiomas representados por ela. Desta forma, o serviço tem a garantia de que a informação passada para ele possui a semântica necessária para a execução correta do serviço.

Como exemplo de validação, podemos citar a questão dos horários de início e fim. Se uma dada ontologia representa um compromisso com data final anterior à data inicial, o provedor de serviço será capaz de detectar essa inconsistência e lançar uma exceção antes de invocar o serviço, garantindo que os dados acessados por este serviço permaneçam consistentes.

A adoção de ontologias para descrição semântica dos calendários também é uma alternativa interessante para facilitar a integração de diferentes sistemas de calendário. Já existem esforços e comitês para a criação de ontologias padrão para a representação de calendários. Com um número reduzido de formas de representar calendários, torna-se uma tarefa trivial construir um conjunto de serviços de tradução capazes de converter representações de calendário de uma ontologia para outra.

4.3. Resumo dos Estudos de Caso

A construção dos dois estudos de caso permitiu uma compreensão maior da estrutura de aplicações orientadas a serviços e de como os mecanismos da Web semântica podem afetar o seu desenvolvimento.

No contexto das duas aplicações, as diferenças entre os dois domínios permitiram o levantamento de características que são importantes para as aplicações baseadas em serviços.

Um das principais diferenças é a forma de representar o conhecimento referente ao domínio da aplicação. Enquanto a aplicação de busca precisa tratar da pluralidade de maneiras de representar o conhecimento, no domínio da aplicação de calendários existe um conjunto bem definido de formas diferentes de representar as informações.

No segundo caso, a abordagem de tradução é a mais facilmente aplicada, uma vez que os padrões são conhecidos e eventuais diferenças podem ser contornadas dentro dos serviços de tradução. Para aplicações semelhantes à busca, onde existe um número indefinido de maneiras de representar a informação, é difícil criar serviços de tradução, sobretudo porque o número de serviços de tradução cresce de forma proporcional ao quadrado do número de representações diferentes. Por exemplo, se existirem dez formas de representação diferentes, serão necessários noventa serviços de tradução para permitir completa integração entre todas as formas.

Neste caso, a melhor alternativa é a criação de uma ontologia centralizada de integração, que possui mapeamentos para o conjunto de representações em questão. Essa ontologia pode ser encarada como uma representação canônica, para a qual todos as outras podem ser mapeadas.

Enquanto o uso de serviços de tradução permite uma coordenação de dados mais flexível, a representação canônica traz a necessidade de mecanismos mais

complexos. A abordagem de serviços de tradução pode realizar a integração dos serviços de informação em tempo de execução, sem saber a priori os diferentes tipos de representação. Por outro lado, na segunda abordagem, é necessário que exista um mapeamento entre cada representação e a ontologia de composição.

A partir destas constatações, percebeu-se a necessidade da criação de um conjunto de ferramentas e de uma arquitetura genérica para a criação de aplicações baseadas em serviços que levassem em consideração todas as questões levantadas durante o desenvolvimento dos estudos de caso.

Estas ferramentas deveriam acelerar o desenvolvimento das aplicações, livrando os programadores de tarefas recorrentes na programação da coordenação tanto funcional quanto de dados de serviços distribuídos, além de oferecer suporte aos mecanismos oferecidos pela Web Semântica para o desenvolvimento de aplicações, sobretudo ontologias, como forma de fornecer semântica aos dados trocados no contexto destas aplicações. Também faz-se necessário o suporte a ambientes com disponibilidade limitada na rede, como PCs comuns e telefones celulares, uma vez que estas aplicações poderão utilizar serviços disponibilizados por um enorme gama de dispositivos de acesso diferentes.

Cabe ao desenvolvedor analisar o conjunto das ferramentas e determinar quais se aplicam melhor para o desenvolvimento de uma aplicação baseada em serviços específica. Estas ferramentas podem ser combinadas e estendidas de forma a diminuir o tempo de desenvolvimento e a complexidade das aplicações envolvidas.

Finalmente, todas estas ferramentas devem estar integradas em uma arquitetura genérica para aplicações baseadas em serviços, que pode então ser utilizada em qualquer projeto, facilitando a concepção da aplicação como um todo.