OPERATIONAL AGILITY

COMPOSING AND ORCHESTRATING MISSION CAPABILITY PACKAGES THROUGH BUSINESS PROCESS EXECUTION LANGUAGE (BPEL)

Gary R. Shaffer

Center for Advanced Information Technology,
Science Applications International Corporation
4161 Campus Point Court
San Diego, CA 92121

Phone: 858-826-5746
Fax: 858-826-5617

gary.r.shaffer@saic.com

Report Documentation Page				*Form Approved* *OMB No. 0704-0188*
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.				
1. REPORT DATE **JUN 2004**	2. REPORT TYPE			3. DATES COVERED **00-00-2004 to 00-00-2004**
4. TITLE AND SUBTITLE **Operational Agility. Composing and Orchestrating Mission Capability Packages Through Business Process Execution Language (BPEL)**				5a. CONTRACT NUMBER
				5b. GRANT NUMBER
				5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)				5d. PROJECT NUMBER
				5e. TASK NUMBER
				5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Science Applications International Corporation ,Center for Advanced Information Technology,4161 Campus Point Court,San Diego,CA,92121**				8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR’S ACRONYM(S)
				11. SPONSOR/MONITOR’S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**				
13. SUPPLEMENTARY NOTES **The original document contains color images.**				
14. ABSTRACT				
15. SUBJECT TERMS				
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES **41**	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**			

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Abstract

One aspect of the Department of Defenses' vision for Net-Centric Operations and Warfare is composing and orchestrating Mission Capability Packages from various disparate and geographically dispersed web services into mission-oriented application as required by the operational situation. This allows mission-oriented capabilities to be quickly composed in response to new challenges, requirements, or demands. In other words, *Operational Agility*. Today, web services can communicate with each other, advertise themselves, and be discovered and invoked using industry-wide specification. However, until recently, orchestrating these fine grained services together into coherent course grained solutions required non standard methods and procedures that were generally not interoperable with other organizations. Business Process Execution Language (BPEL) for Web Services (BPEL4WS) mitigates the issue of interoperability by providing a set of constructs, based on XML, that can be used to define the semantics of how process communicate and exchange data, control the flow of data from one services to another, and the order in which to invoke services. Furthermore, subject matter experts using graphical designer tools and not software developers writing software components can compose the processes. This will allow mission-oriented capabilities to be quickly composed in response to new challenges, requirements, or demands.

"…leveraging information technology and innovative network-centric concepts of operations to develop increasingly capable joint forces. Our ability to leverage the power of information and networks will be key to our success…"

Deputy Secretary of Defense Paul Wolfowitz

Introduction

To implement Secretary Wolfowitz's vision, what is needed is greater horizontal integration focused on warfighting capabilities. Currently, our military's C4ISR infrastructure suffers from highly stove-piped systems and integration that is at best vertically focused along the functional lines of Intelligence, Surveillance, and Reconnaissance (ISR), Command and Control (C2), and Fire Control (FC). Under the transformational initiative called FORCEnet, the Navy has defined the notion of Engagement Packs. FORCEnet Engagement Packs (FnEP) [1][2] are an "alignment tool" that focuses on improving the speed and the effectiveness of command and control, improving tactical force integration, and expanding combat reach through real-time orchestration of engagement assets. Specifically, the development of FnEPs accelerates the development and "*operationalization*" of FORCEnet and will allow the Navy to realize FORCEnet operational capabilities by achieving flexible, adaptable, and net-centric warfighting capabilities to address both conventional and asymmetric threats.

FnEPs create a construct for integrating Intelligence, Surveillance, and Reconnaissance Command and Fire Control information exchange requirements across the engagement chain. FnEPs are the operational construct for how the Navy will make FORCEnet a reality. The concept integrates FORCEnet factors (sensors, networks, warriors, command and control, platforms and weapons) to yield Combat Reach Capabilities (CRCs) which are, in turn, combined to form packs – combinations of naval and joint assets used to engage the enemy.

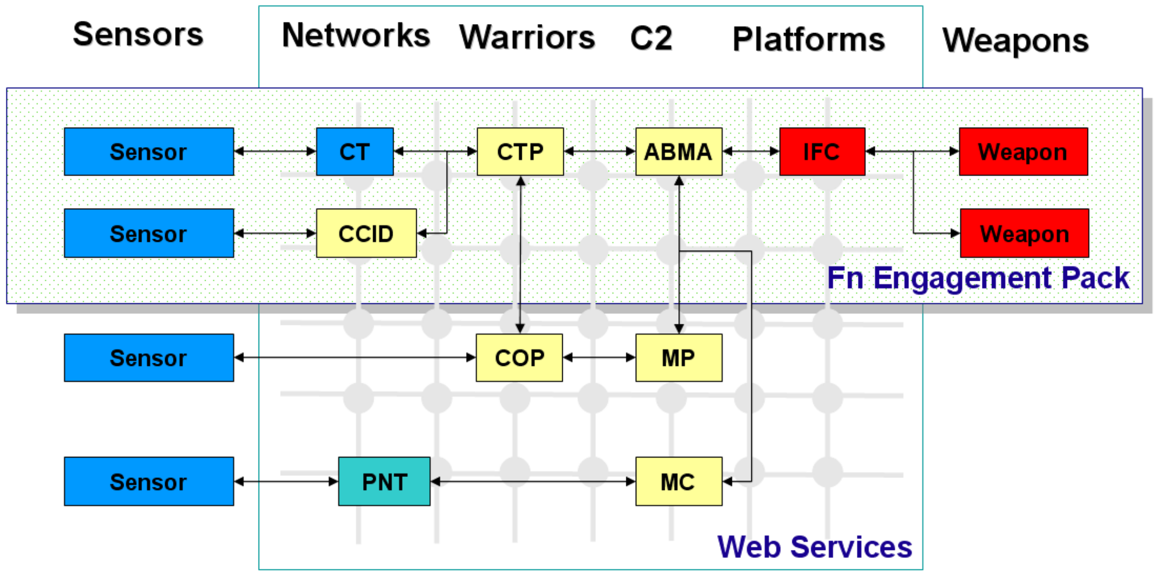


Figure 1. FORCEnet Engagement Packs [3]

As depicted in Figure 1, FnEPs integrate command and control, sensor information, and fire control systems into a mission specific engagement chain. This increases the Navy's offensive and defensive combat reach, reduces timelines, and reduces integration and interoperability problems. FnEPs accomplish this by optimizing distributed sensor-weapons-target pairing linkages such that the Navy will have the ability to implement new innovative concepts with existing Naval and Joint systems. It is envisioned that each of these capability (Integrated Fire Control (IFC), Common Operational Picture (COP), Mission Planning (MP), Composite Combat Identification (CCID), etc.) will be composed of tens, if not hundreds or thousands of disparate and distributed visualization and data oriented web services and authoritative data sources. It is entirely possible that aggregations will occur on multiple levels (aggregates of aggregates of aggregates…).

Web Services are self-contained, modular business process applications that are based on the industry-standard technologies such as eXtensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), and Universal Description Discovery and Integration (UDDI). Their purpose is to enable users to connect different components in a platform- and language-independent manner across organizational boundaries. However, none of these standards allow defining the business or process semantics of Web Services. In other words, they provide no semantics for linking of services together into a coherent operational process or composition. Thus, today's Web Services are isolated and opaque. Breaking isolation means connecting services and specifying how collections of services are jointly used to realize more complex functionality. A process specifies the potential execution order of operations from a collection of services, the data shared between these services, the transactional states, which partners are involved and how they are involved in the process, joint exception handling for collections of services, and other issues involving how multiple services and organizations participate.

The first question one must ask is how does one provide the ability to dynamically re-configure and re-allocated assets "on-the-fly" to re-constitute a new capability due to changing mission requirements in a way that supports cross-mission engagements?

Technology is a good thing but by itself, doesn't solve the needs of the DoD nor does it necessarily help its deployed forces accomplish their mission. It can be argued that in many ways, technology even gets in the way. In some ways, it can be said that Microsoft's PowerPoint is the most widely use mission-planning software in the world. The challenges of building a commanders briefing collaboratively, across components distributed from a Unified Combatant Commands HQ to ones in the AOR are well known. A tremendous amount of time and energy is expended nightly in building the daily situation briefing for commanders such as USCENTCOM's General Abizaid.

Each directorate is responsible for its portion of the briefing. It is usually performed by each JOC watch officers and several subordinates and consumes anywhere from one to four hours for each person involved. For example, The J2 (INTEL) watch office and his staff are responsible for building out the intelligence portion of the briefing. This includes collecting intelligence summaries, imagery from satellites, intercepts, annotations, estimates, etc., into one or more slides. Another example is that of the METOC watch officer. This watch stander is responsible for developing slides which detail the effect of current and forecasted weather on the planned operations for the coming days. When the weather effects slides are completed, they are incorporated into the J3 (OPS) set which is then combined with the other directorates' slides into a complete briefing for approval by the JOC Chief.

On a recent visit to COMTHIRDFLT an officer, who regularly pulls duty as the METOC watch officer, was asked what steps he went though in preparing his inputs into the commanders daily briefing. What he told us was eye opening! He started out by stating that he had a loose leaf binder in his office that contained all of the notes and/or instructions on how to compute the effects of weather on operations. These have been given to him by the officer that he replaced and have been modified over time. More importantly, they are not used by other watch standards in his or other commands.

The second question one must ask how can technology help reduce the workload placed on our armed forces in a way that allows the warfighter (subject matter experts), not programmers, to take advantage of them to accomplish their duties?

The third question one must ask is how can technology capture corporate knowledge before the person who has that corporate knowledge moves on to better and brighter things?

Hopefully, the remainder of this paper will answer these two questions.

Business Process Execution Language

Business Process Execution Language (BPEL) for Web Services (BPEL4WS) represents the merging of IBM's Web Services Flow Language (WSFL) [4], which provides support for graph-oriented processes, and Microsoft's web services for business process design (XLANG) [5], which is based on structural constructs for business processes. In May 2003, the second release of the specification (version 1.1) authored by IBM, Microsoft, BEA, SAP, and Siebel Systems was ratified by Organization for the Advancement of Structured Information Standards (OASIS) [6] and software vendors such as SUN, Oracle, BEA, IBM, Microsoft, and others have all indicated that support for BPEL will be included in future releases of their respective products.

BPEL allows specification of processes and how they relate to services. This includes specifying how a process makes use of services to achieve its goal, as well as specifying services that are provided by a given process, procedures, policy or doctrines. It specifies the potential execution order of operations from a collection of services, the partnerships required and joint exception handling for collections of web services. Long running transactions can be specified between sets of web services thus increasing reliability and consistency for services. Processes specified via BPEL also prescribe the exchange of messages between services that compose the process. These messages are WSDL messages of operations of the port types involved in the roles of the service links established between the process and its partners. They are fully executable and portable between BPEL-conformant environments. A BPEL process interoperates with the services of its partners, whether or not these services are implemented based on BPEL. Finally, BPEL supports the specification of business protocols between partners and views on complex internal operational processes.

Because each BPEL is a web service, a BPEL can orchestrate the interaction and data flow between other BPELS as well, thus providing the ability to build more complex automated workflows while preserving data encapsulation [7].

The general relationship between the standards involved is summarized in Figure 2. BPEL uses WSDL to specify actions that take place in a process and to describe the services provided by a process. WS-Transaction [14] specifies a protocol for the long-running transaction model defined in BPEL as well as atomic transactions between regular services. Applications created with BPEL are called *process-based applications*. This kind of application structure splits an application into two strictly separated layers:

the top layer, or process, is written in BPEL and represents the flow logic of the application. The bottom layer, or services, represents the function logic of the application.

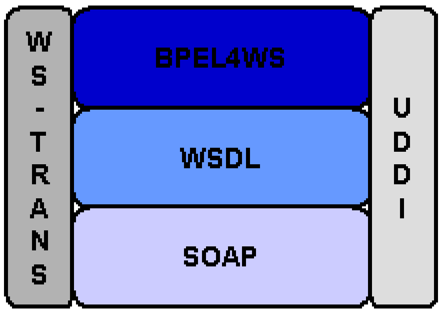


Figure 2. BPEL4WS' relationship to other standards [8]

This layering has several advantages over more conventional approaches. Because they have been separated, either one can be changed without any impact on the other services within the application or on the services that the process represents. In addition, the application can be developed and tested in two separate stages: the process can be developed and tested separately from the development and testing of the individual services. This approach provides great flexibility in evolving an application in an environment in which incremental development takes place.

In a BPEL process, everything is XML, including the messages that are passed into and returned from the BPEL process, the messages that are exchanged with external services, and any local variables used by the flow itself. The types for all of these messages and variables are defined with XML Schema, usually in the WSDL file for the flow itself or in the WSDL files for services it invokes. Therefore, all variables in BPEL are XML documents, and any interesting BPEL process will spend a fair amount of its code manipulating those XML variables.

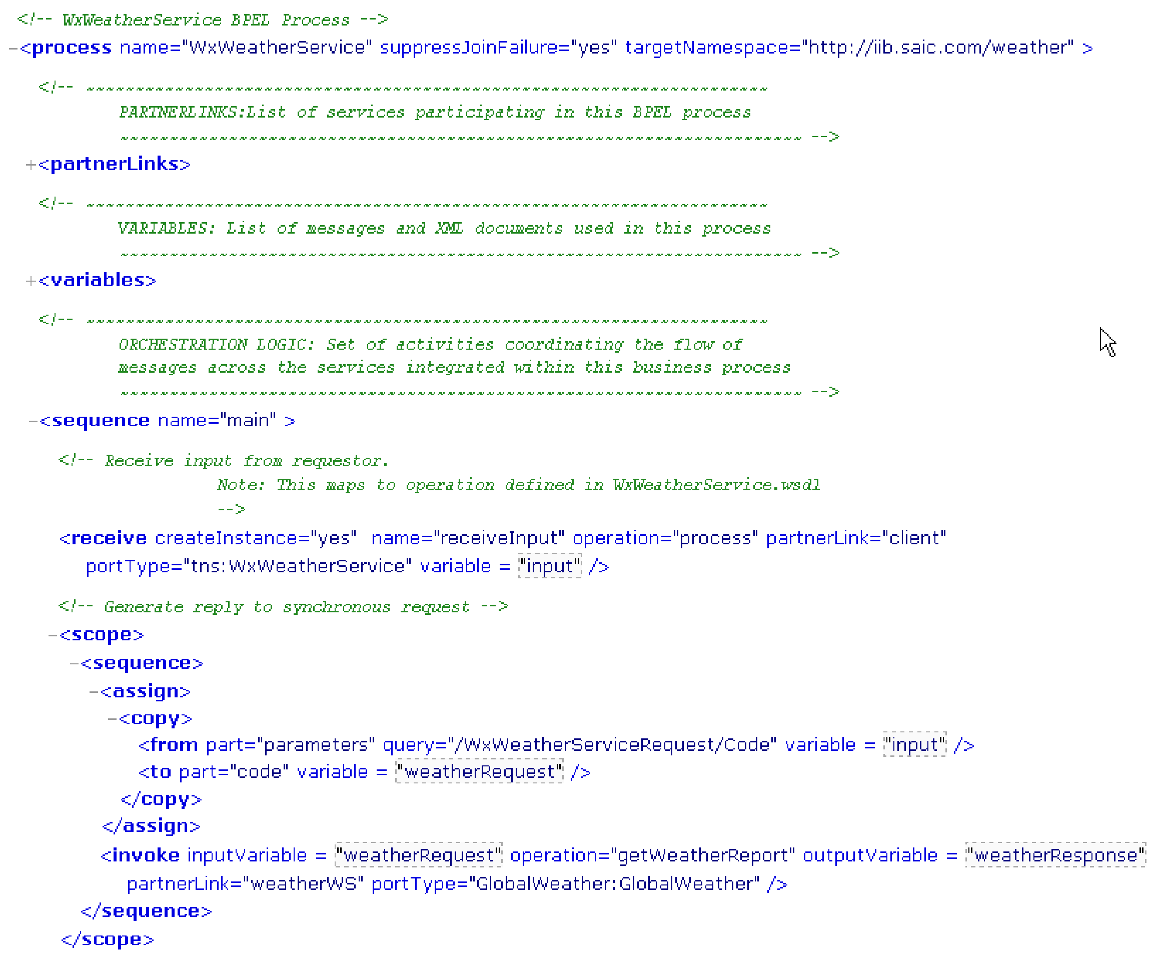
Conceptually, processes written in BPEL are "flow-chart" expressions of an algorithm. A process is decomposed into discrete steps, known as "activities" and is associated with a specific XML tag:

- <*invoke*> is used to invoke an operation on a given web service
- <*receive*> is used to wait for some external web service to invoke a given operation of the process
- <*reply*> is used to generate a response to an input/output operation
- <*assign*> is used to copy data from one place to another
- <*wait*> is used to wait for some specific period of time
- <*throw*> is used to signal that some error condition has occurred
- <*terminate*> is used to terminate a process instance
- <*empty*> simply does nothing

More complex expressions of functionality can be achieved by aggregating primitive activities using the any of the “structure” activities:

- <sequence> provides the capability to define an ordered sequence of steps
- <while> provides the capability to loop until some specified criteria is met
- <switch> is analogous to the case statement found in most modern programming languages
- <flow> indicates that collections of steps are executed in parallel
- <pick> provides the ability to execute one of several alternate paths within a process flow

Listing 1 depicts a portion of a BPEL in its XML form.



```
 <!-- WxWeatherService BPEL Process -->
-<process name="WxWeatherService" suppressJoinFailure="yes" targetNamespace="http://iib.saic.com/weather" >

   <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
          PARTNERLINKS:List of services participating in this BPEL process
          ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
  +<partnerLinks>

   <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
          VARIABLES: List of messages and XML documents used in this process
          ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
  +<variables>

   <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
          ORCHESTRATION LOGIC: Set of activities coordinating the flow of
          messages across the services integrated within this business process
          ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
  -<sequence name="main" >

     <!-- Receive input from requestor.
                    Note: This maps to operation defined in WxWeatherService.wsdl
                    -->
     <receive createInstance="yes"  name="receiveInput" operation="process" partnerLink="client"
       portType="tns:WxWeatherService" variable = "input" />

     <!-- Generate reply to synchronous request -->
    -<scope>
      -<sequence>
        -<assign>
          -<copy>
             <from part="parameters" query="/WxWeatherServiceRequest/Code" variable = "input" />
             <to part="code" variable = "weatherRequest" />
           </copy>
         </assign>
         <invoke inputVariable = "weatherRequest" operation="getWeatherReport" outputVariable = "weatherResponse"
           partnerLink="weatherWS" portType="GlobalWeather:GlobalWeather" />
       </sequence>
     </scope>
```

Listing 1. BPEL Source for Operational Process

BPEL also allows one to recursively combine the structured activities to express arbitrarily complex algorithms that represent the implementation of a service. Once deployed, BPEL processes are exposed to other organizations as web services to be consumed as normal web services using the “find”, “bind”, and “execute” paradigm.

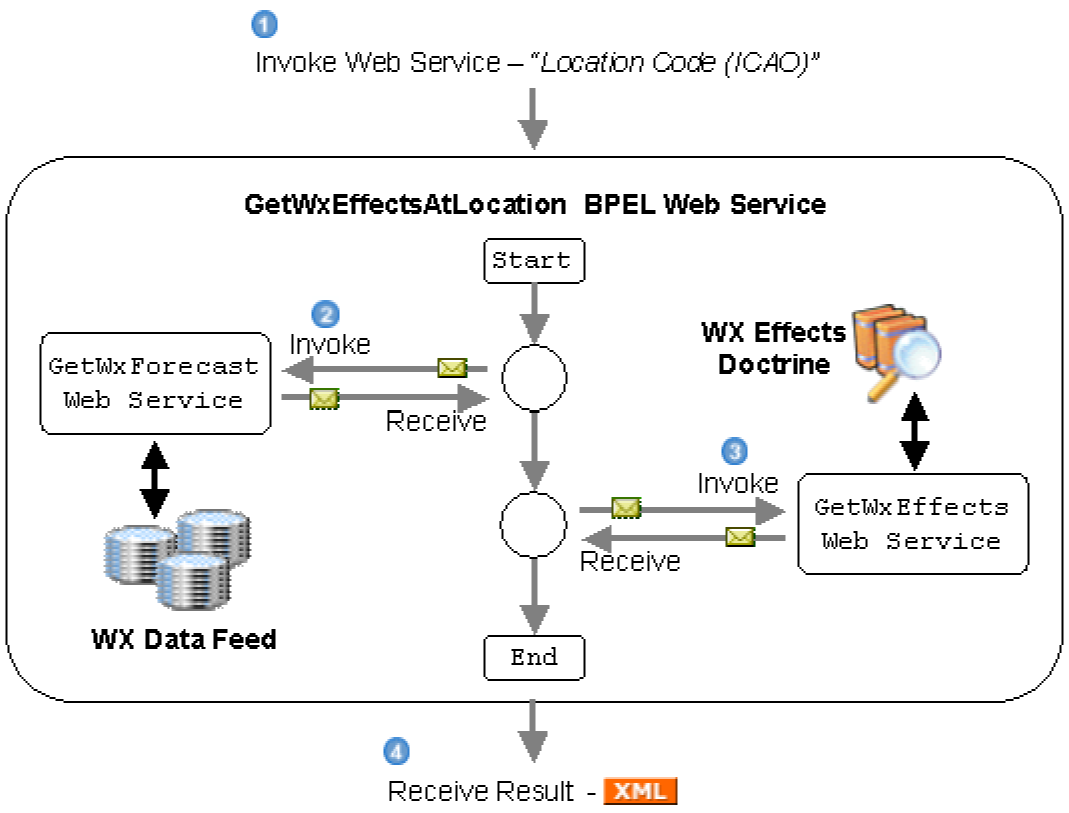


Figure 3. An Example Operational Process

Figure 3 depicts an example operational process that could be to determine the effects of weather on operations for a specified location. Input for this process flow is the International Civil Aviation Organization (ICAO) 4-letter airport identifier code. The process itself is a single thread that is instantiated by some caller via the <receive> tag. It then makes two synchronous calls (using <invoke>) to other external web services. The first one to the "*GetWxForcast*" web service obtains the current weather conditions for the specified ICAO. These calls are synchronous in that they wait for a reply message from the invoked web service before they continue. Asynchronous calls do not wait for a reply message. Instead they setup a rendezvous using another <receive> tag. The second call is to the "*GetWxEffects*" web service. This web service, using the weather data provided from the "*GetWxForcast*" service, computes the effects of weather on operations based on field manual 'FM 34-81-1' (Weather effects on Operations). The data retuned by the second web service are coalesced with the weather forecast from the first and are returned to the original caller as XML using the <reply> construct.

Building Operational Processes

The power of BPEL is that one doesn't have to be a "rocket scientist" to implement operational processes in BPEL. A basic understanding of the nomenclature and standards (i.e., what is WSDL), where to find web services (a UDDI) and the capabilities they provide is all that is needed. Through the use of a graphical BPEL design tool, Subject

Matter Experts (SMEs) expose external web services then drag-n-drop these web services along with BPEL actions onto the designer window to build a process flow diagram. Figure 4 depicts screenshot of Collaxa's graphical BPEL Designer tool [9].

As depicted, a web service can be placed in the process flow from the list on the left of the screen. An SME would then drag desired web service actions from the right onto the process flow to define how data will be pulled, aggregated, and transformed. Each action has requisite properties (input/out variables) that can be defined. On the left is the client interface, displaying operations exposed by the process (initiate in this case) and asynchronous callback operations (onResult). In the middle of the window is a link to edit the Process Map for the flow (visual representation of the flow logic) and any global XML variables defined for the flow ("input" and "output" in this case, which are created automatically by the New Project wizard). An Inspector pane will be shown on the right once you drill down into the process map view.

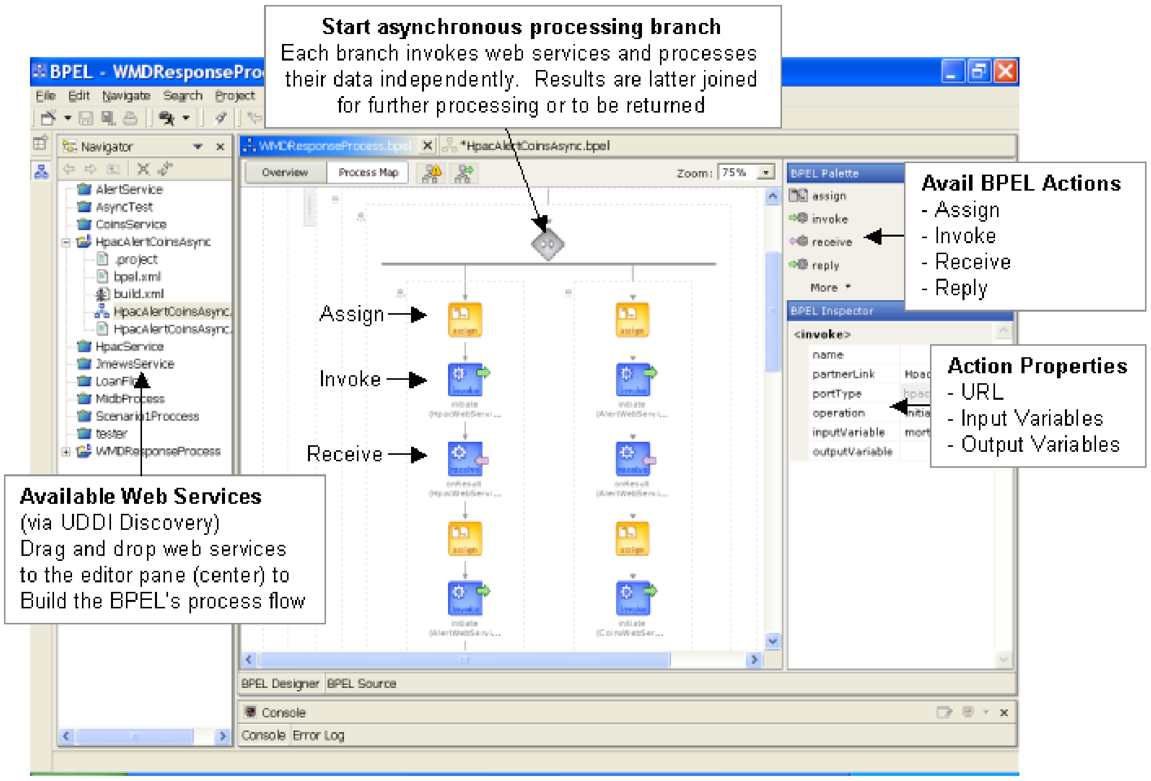


Figure 4. Subject Mater Experts Build Operational Processes using BPEL

Once the SME has finished building the process flow, the BPEL Designer can auto-generate the new BPEL web service. The tool will also build the new web services WSDL. The last step is for the SME to register (deploy) the new web service. This will load it into the BPEL execution engine and make it and its WSDL available throughout the enterprise. At this point, it can be used like any other web service. It can be called by other external web services, even other BPEL processes. It can be combined (aggregated) with other BPEL processes to form more complex functionality.

UML, BPEL, and Model Driven Architectures

The Unified modeling Language (UML) is a language for modeling systems and is the most widely known Object-Oriented modeling notation in use today. It has a graphical notation which is readily understood, and contains a rich set of semantics for capturing key features of object-oriented systems. UML is widely used in the development of object-oriented software and has also been used, with customizations, for component-based software, business process modeling, and systems design. This enables the considerable body of UML experience to be applied to maturing Web services technologies.

The ability to extend UML is essential to this concept. UML can be customized to support the modeling of systems that will be completely or partially deployed to a Web services infrastructure. This approach enables BPEL based operational processes to be incorporated into an overall system design utilizing existing software engineering practices. Additionally, the mapping from UML to BPEL4WS permits a model-driven development approach in which executable operational processes can be automatically generated from UML models.

IBM's work in this area is centered on the use of Activity Diagrams and Stereotypes as a way of categorizing elements of a model [11]. For instance, if you have a class representing a "*Target*", you could attach a stereotype of <<entity>> to indicate that it represents a data object. This information is used it to guide the behavior of a model translator. Stereotypes can be added to most elements in a UML model. A UML Profile is used to define a specific set of extensions (the BPEL Stereotypes) to the base UML in order to represent a particular domain of interest. Using this concept, processes are represented as a class with the stereotype <<Process>>. The attributes of the class correspond to the state of the process (variables in BPEL). Table 1 maps the UML to its corresponding notion in BPEL.

Table 1. UML to BPEL mapping

UML Stereotype	**BPEL**
<<process>> class	BPEL process definition
Activity graph on a <<process>> class	BPEL activity hierarchy
<<process>> class attributes	BPEL variables
Hierarchical structure and control flow	BPEL sequence and flow activities
<<receive>>, <<reply>>, <<invoke>> activities	BPEL activities

Activity graphs are used to describe the behavior of a class. The actions to be performed are shown as Entry conditions to the activity; for example, Track ID (a variable) is set to the result of the correlation service. The partners with which the process communicates are represented by the UML partitions (also known as swimlanes): Sensor, Correlation, TDBM, and COP. Activities that involve a message send or receive operation to a partner appear in the corresponding partition. The arrows indicate the order in which the process performs the activities. This is depicted in Figure 5.

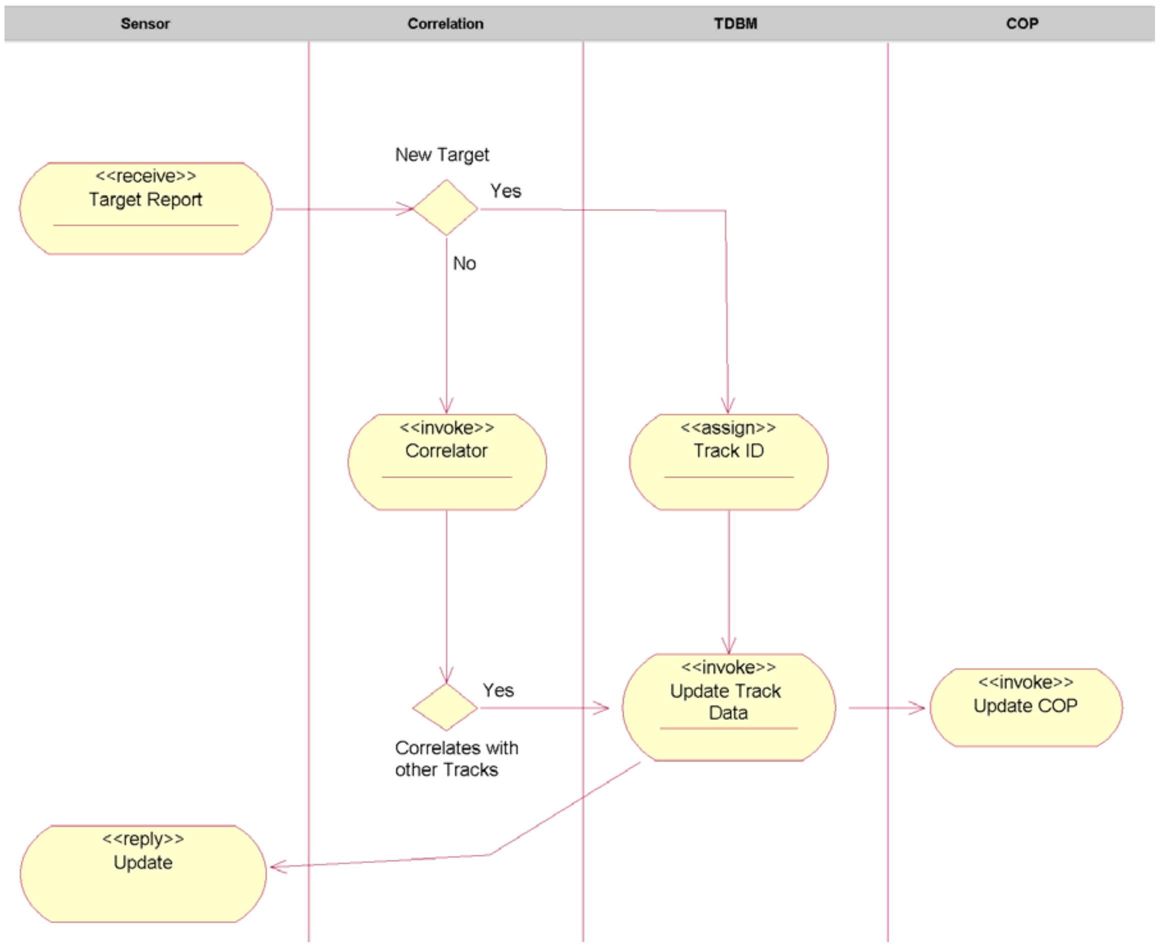


Figure 5. BPEL Process as a UML Activity Diagram

Using current UML development tools, such as Rational's XDE, or ArgoUML, to take models of processes and convert them into their corresponding BPEL and WSDL files necessary to implement that process is a powerful argument for Model Driven Architectures (MDA) as proposed by the Object Management Groups (OMG) [10]. MDAs aims to raise the level of abstraction at which development occurs which in turn, delivers greater productivity, better quality, and insulation from underlying changes in technology.

SPAWAR Distributed Services Commercial Area Announcement

In response to SPAWAR's Distributed Services Commercial Area Announcement (CAA) [12], SAIC and its partners defined a *System of Systems* based on a Service Oriented Architecture that promulgated the tenets of an Enterprise Services Bus (ESB). Our approach incorporated a set of lightweight, Distributive Service centric components that were application-server-independent. It offered up a set of Core, Enterprise, and Mission specific services such as:

- Authentication and Authorization Services which isolated our Identity Management solution from the bus.
- A Registry Service for registering distributed services.
- A Messaging/Alert Service which implemented the Publish/Subscribe paradigm for information flow.
- An Orchestration Service based on a commercially available BPEL server from Collaxa.
- Commercial and Open Source portal products.
- A set of legacy/mission applications comprising mission planning, medical, strike packages, hazardous plume analysis, logistics, and weather.
- A set of authoritative data source for medical, blue and blue forces, intelligence, and weather.

The architectural vision for this bus is the FORCEnet Government Reference Architecture (GRA) [13] which promulgates *Dynamic Force Composition*.

One of the appeals of using an ESB is that MCPs are built purely in terms of these services, and the exact location and implementation details of these services are transparent to the warfigther. This location transparency gives the engineers a lot of flexibility in terms of where they host services and also allows services to be relocated at a later stage without disruption to existing MCPs.

Our focus was on providing an architectural framework that allows for the decoupling of interface objects from implementation objects and exposing interfaces to enable third-party integration with that object code. Emphasis was placed on platform-, system-, and vendor-independence in a distributed environment to obtain the greatest dissemination of knowledge in the shortest period of time, thereby providing the warfighter with an agile system that can adapt to the dynamics of the current operation situation.

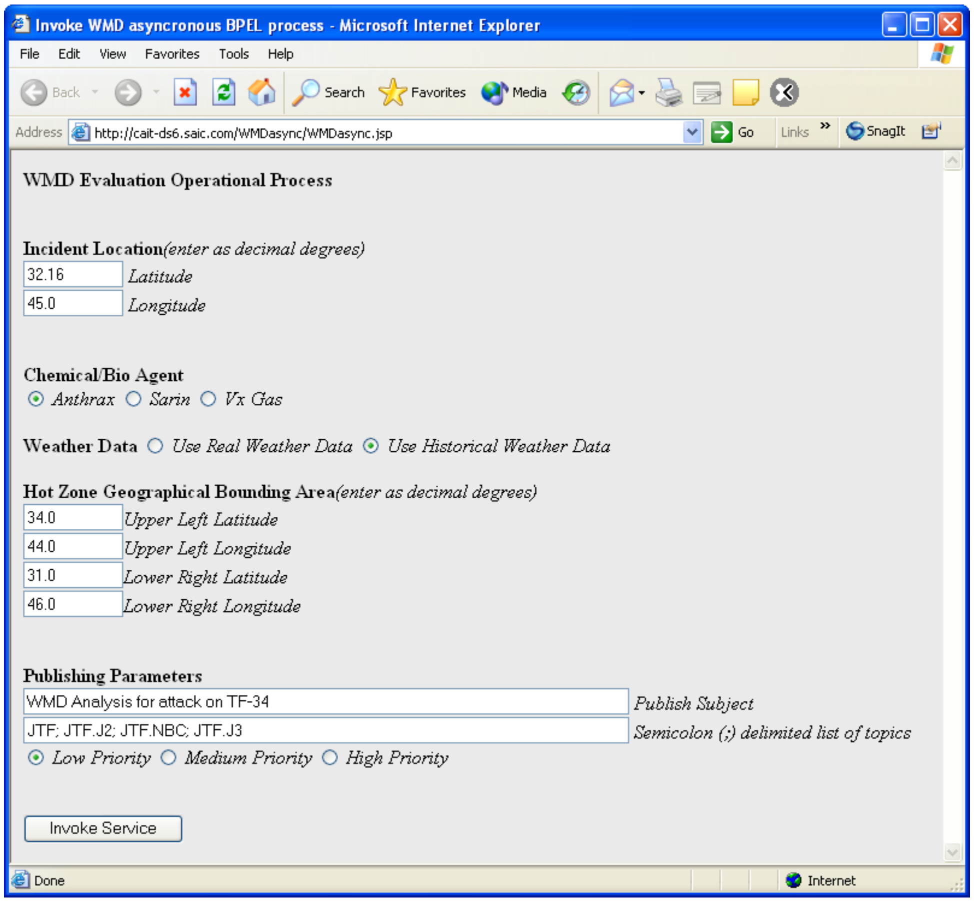
Our implementation of the ESB does not require all existing applications from Global Command and Control System – Maritime (GCCS-M) and/or other candidate systems to be rewritten as services; however, any common service provided by those applications should be exposed as a Service using open standards and distributed services technologies. This means a Service can be implemented as a simple Service wrapper (façade), translating the Service interface calls into the application-specific access methods and business logic. The ability to integrate legacy systems in this fashion preserves prior investments and leverages knowledge gained to enhance the systems' value by increasing the user base and enabling shared business processes and data.

To demonstrate the viability of this approach, the team developed a scenario in which the services and their respective underlying legacy mission applications could sense, fuse, and present actionable knowledge to the warfighter. The scenario for this demonstration was centered on the release of anthrax against friendly U.S. ground forces by a rogue battalion-sized artillery group.

Operational Process Example: WMDAnalysis

The scenario used to demonstrate the Distributed Services developed for the SPAWAR 05 CAA used a fairly elaborate Warfighter's business process developed around the idea of automatically responding to alerts generated by WMD sensor alarms. Core Services, legacy mission applications exposed as web services and authoritative data sources where brought together as a System of Systems that could sense, fuse, and present actionable knowledge to the decision maker. In this instance, the watch standers of a JTF stood up in support of Operation Iraqi Freedom (OIF). The goals of this operational process are:

1. Initiate hazardous plume analysis.
2. Notification of all deployed units within Area Of Interest (AOI) that a release has occurred so that they can take appropriate action such as raising the MOPP status or moving upwind, etc.
3. Provide relevant medical information to downstream medical facilities and personnel with respect to casualty data.
4. Start gathering information/knowledge for the JTF's Joint Planning Group (JPG) so they can start Course Of Action (COA) planning immediately upon activation.


Invoke WMD asyncronous BPEL process - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://cait-ds6.saic.com/WMDasync/WMDasync.jsp
Go Links SnagIt

WMD Evaluation Operational Process

Incident Location*(enter as decimal degrees)*

32.16 *Latitude*

45.0 *Longitude*

Chemical/Bio Agent

[✓ *Anthrax*] *Sarin* *Vx Gas*

Weather Data *Use Real Weather Data* [✓ *Use Historical Weather Data*]

Hot Zone Geographical Bounding Area*(enter as decimal degrees)*

34.0 *Upper Left Latitude*

44.0 *Upper Left Longitude*

31.0 *Lower Right Latitude*

46.0 *Lower Right Longitude*

Publishing Parameters

WMD Analysis for attack on TF-34 *Publish Subject*

JTF; JTF.J2; JTF.NBC; JTF.J3 *Semicolon (;) delimited list of topics*

[✓ *Low Priority*] *Medium Priority* *High Priority*

Figure 6. WMD Analysis Portlet

For the DS CAA demonstration, the portlet depicted in Figure 6, was presented to the operator and initiates the BPEL process (this could have been automated this by having the underlying BPEL process subscribe to several message queues to collect all of the requisite information). This portlet invoked an asynchronous BPEL process that was aggregated from several other asynchronous BPEL processes. This process is depicted in Figure 7 (for this paper, the diagram has been collapse to hide most of the programmatic details).

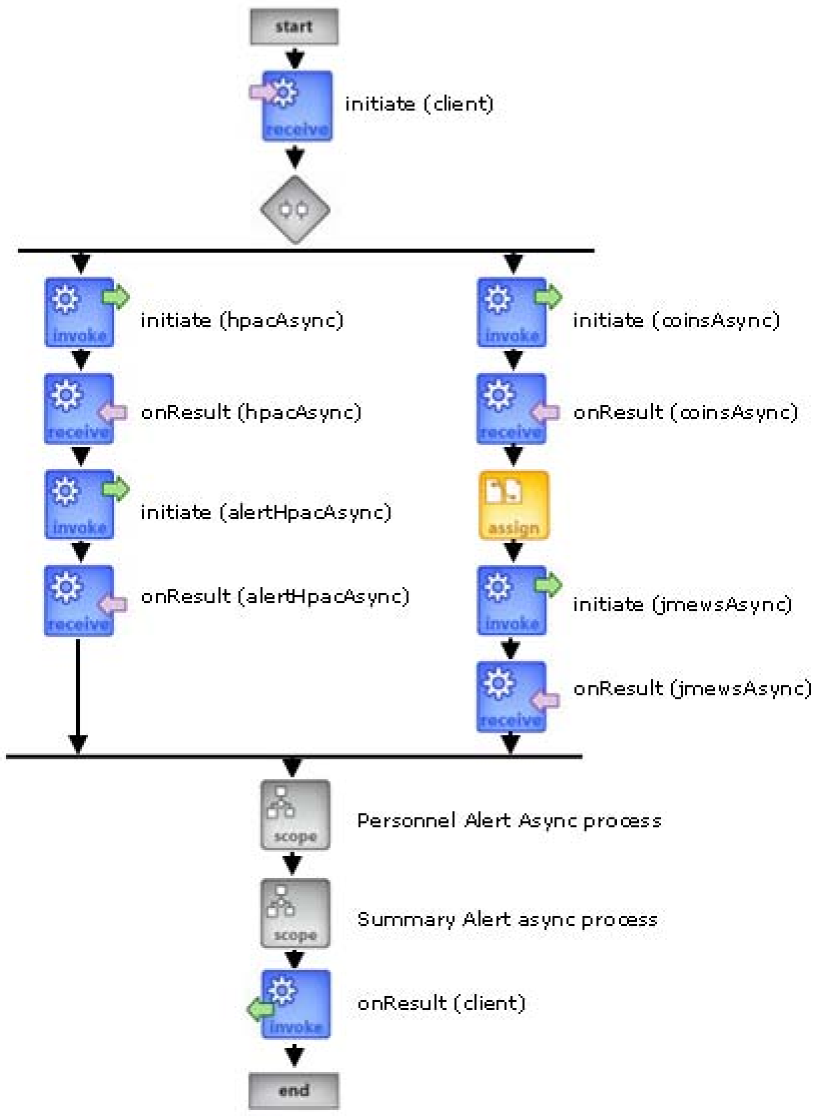


Figure 7. BPEL Process Diagram for WMD Analysis

After initialization, the process branches (using the <flow> tag) into two parallel processes. It is also important to note that this operational process orchestrates other BPEL processes that actually call the required web services and allow their results to be consumed by other processes in the flow in an asynchronous manor. One branch invokes the "*hpacAsync*" BPEL process which in turn invokes the Hazardous Plume Analysis Capability (HPAC) Web Service (initiate (*hpacAsync*)). This initiates the development of a "Probability of Mortality" plume overlay for display on a COP. It also provides a casualty estimate for the indigenous population. Once the plume data is returned (onResult (*hpacAsync*)), the branch them invokes the "*alertHpacAsync*" BPEL process which in turn invokes the Alert Service Web Service (initiate (*alertHpacAsync*)) which publishes the plume data on the supplied message queues. Once the alert has been published, this branch waits to rendezvous with the second branch.

The second branch invokes the "*coinsAsync*" BPEL process which invokes the Coalition Interoperability System (COINS) developed for CINC-21 to obtain a list of all coalition personnel within the AOI. The "assign" step extracts personnel identifiers from the resultant XML data. This data is then fed into the "*jmewsAsync*" BPEL process which invokes the Joint Medical WorkStation (JMeWS) Web Services. This authoritative data source uses the data to compute an initial list of personnel who have been/not been vaccinated against Anthrax. On return from the "*jmewsAsync*" process, the second branch rendezvous with the first branch. It then uses the data collected from both branches to compute a preliminary detailed coalition casualty estimate and publish it. Finally, it aggregates the data from HPAC about the expected indigenous population casualties with the coalition and blue forces casualty information to provide a complete casualty summary which is them published to the Alert Service for consumption by others.

Several other BPEL processes were developed in support of COA development:

- Invoke the JMeWS web service to obtain a list of available medical treatment facilities within the AOR.
- Invoke the MIDB web service to obtain a list of airfields within a radius of 125 miles of the AOI.
- Invoke the JMeWS web services to obtain a list of available decontamination and medical supplies with the AOR
- Invoke the eNTCSS web service to obtain a list of available operationally ready aircraft (FMC/MC) that could potentially participate in an evacuation. This data was further constrained buy the list of airfields returned by the MIDB web service.
- Invoke the JMeWS and COINS web services to obtain lists of available medical personnel.

Operational Process Example: METOC Watch Officer

The example depicted in the figure below, illustrates a challenge dealt with by JTF staff officers on a daily basis; building the commanders daily situation brief and is based in part from inputs from several officers stationed at COMTHIRDFLT. In gathering data for the various PowerPoint slides, the metrology officer (METOC) performs a repetitive task daily to gather weather forecast data, compare it against defined doctrine, and aggregate results to determine not just the weather conditions in a defined operating area but its effects on operations. For example, if the cloud ceiling is below 5000 m but above 1000m, then fixed wing operations are degraded (yellow status). If the cloud ceiling was below 1000m then fixed wing operations are severely impacted (status red). Calculating these values across the many different mission areas is time consuming, complicated, and error prone.

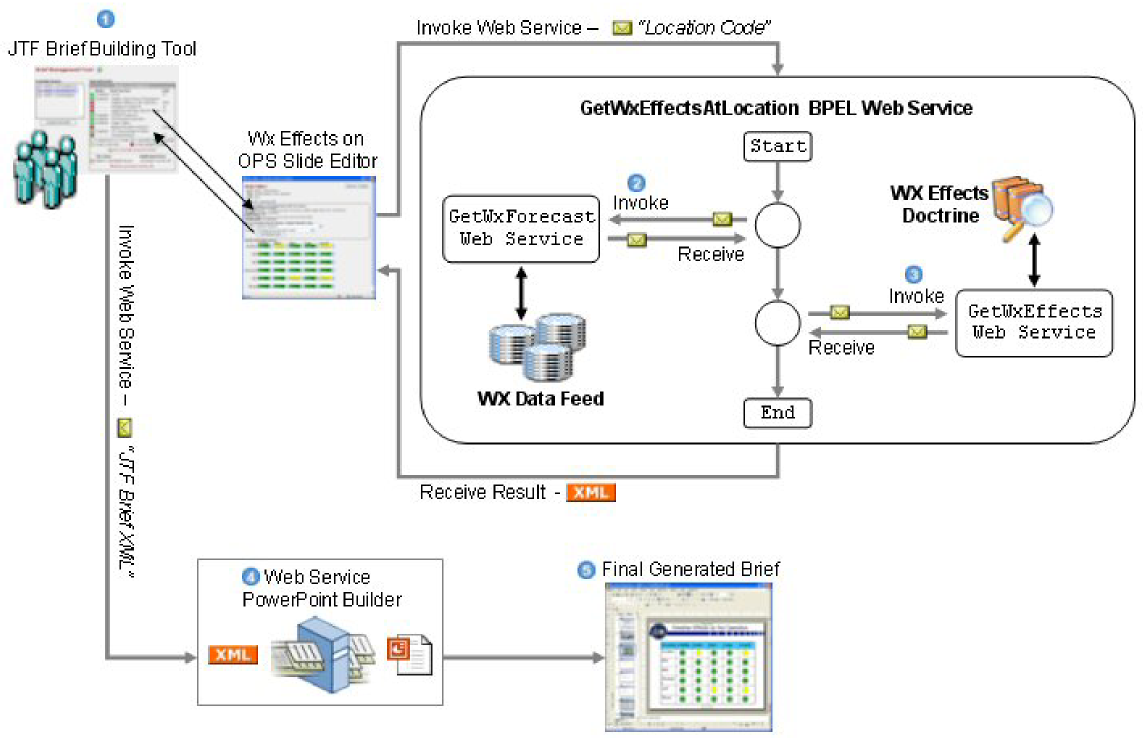

1
JTF Brief Building Tool
Wx Effects on OPS Slide Editor
Invoke Web Service – "Location Code"
GetWxEffectsAtLocation BPEL Web Service
Start
2
Invoke
Receive
GetWxForecast Web Service
WX Data Feed
WX Effects Doctrine
3
Invoke
Receive
GetWxEffects Web Service
End
Receive Result - XML
Invoke Web Service – "JTF Brief XML"
4 Web Service PowerPoint Builder
XML
5 Final Generated Brief

Figure 8. Auto Generating JTF Commander's Update Brief

In the Figure 8, Step #1 represents a web application built to simplify the staff's collaborative challenge of building the brief. In it, there are links for each of applicable slides. It is here the METOC officer selects the "Weather Effects on Operations" slide link. From the METOC officer's view point, a web page displays, the watch officer then selects a location (operating area), and the window is updated with the locations current weather data. Additionally, the browser window displays the correctly updated red/yellow/green stoplight style statuses of operations for each desired mission area (fixed win, Infantry, SOF, and ISR).

Behind the screen a series of actions have been executed. When the location is entered and the get data button is selected on the "Wx Effects on OPS Slide Editor HTML Page", the *GetWxEffectsAtLocation* BPEL process is invoked via an HTTP call. This starts the modeled workflow process, which first invokes a *GetWxForecast* web service (Step #2) to get the current weather conditions at a given location (wind, temp, cloud ceiling, visibility). The resultant XML data is returned to the operational process. Next it invokes another web service, *GetWxEffects* (Step #3) and feeds it the XML output from Step #2 as its input. This web service uses the weather conditions data at a given location to calculate mission status codes for the various mission areas. The business rules for these calculations are governed by doctrine. Most likely, this particular web service has been build and maintained but the organization responsible for METOC doctrine. Again, the output from this web service is sent back to the OP as XML data. The OP then transforms the returned XML data into its WSDL defined output format.

At this point, the METOC officer sees the result of the BPEL process in the form of an updated HTML page. He then reviews the results, adds comments as needed and changes the slides status to complete. The officer in charge of building the PowerPoint brief can now, via the web, view and monitor the status of the slides. This process can be repeated daily, in less time, and accurately. As an added benefit, the same web services can be made available to other commands as well. The METOC operational process provides constancy across the DoD enterprise as well. Its business rules are doctrine driven thus insuring all operational commands throughout the DoD determine mission capabilities in a constant manor. Staff focus has moved from gathering and entering information into a PowerPoint slide to analyzing the effects of the information on executing operational plan and planned missions.

Summary

Under FORCEnet, Operational agility is accomplished by composing, then orchestrating, Mission Capability Packages from various disparate and geographically dispersed services into mission-oriented application as required by the operational situation. BPEL removes the software developer from the loop and hands the role back to subject matter experts who better understand the operational requirements of the current mission, thus allowing mission-oriented capabilities to be quickly composed in response to new challenges, requirements, or demands in ways we have never seen before. The results will be profound. Naval capability packages will be readily assembled from forward-deployed forces. These forces, along with their respective MCPs, will be tailored to meet the mission needs of the Joint Force Commander. They will be sized to the magnitude of the task at hand.

References

[1] Robert Woodrow Hesser and Danny Michael Rieken. FORCEnet Engagement Packs: “Operationalizing” FORCEnet to Deliver Tomorrow’s Naval Network-Centric Combat Reach Capabilities…Today. *Masters Thesis, Naval Postgraduate School*, December 2003.

[2] FORCEnet Engagement Packs are also known as FEP.

[3] Phil Charles. *FEP Analysis Efforts* Briefing. FORCEnet Engagement Packs Workshop, Naval Station Norfolk, Feb 18-19 2004. UNCLASSIFIEID Briefing can be downloaded from https://ekm.netwarcom.navy.mil

[4] http://xml.coverpages.org/wsfl.html

[5] http://xml.coverpages.org/xlang.html

[6] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel

[7] For a description and definition of orchestration, see the Burton Group: Orchestrating Web Services: Driving Distributed Process Execution through Workflow Technology, December 18, 2003.

[8] Frank Leymann and Dieter Roller. *Business Processes in a Web Services World*. IBM, 1 August 2002. http://www-106.ibm.com/developerworks/webservices/library/ws-bpelwp

[9] http://www.collaxa.com/home.index.jsp

[10] http://www.omg.org/mda/index.htm

[11] Keith Mantell, *From UML to BPEL*. IBM, 9 September 2003. http://www-106.ibm.com/developerworks/webservices/library/ws-uml2bpel

[12] This CAA can be downloaded from https://e-commerce.spawar.navy.mil under solicitation number N00039-03-R-0030

[13] Office of the Chief Engineer, SPAWAR 05. FORCEnet Government Reference Architecture (GRA) Vision, version 1.0, 08 April 2003.

[14] http://www-106.ibm.com/developerworks/webservices/library/ws-transpec

Author Bio

Gary R. Shaffer is currently the Chief Technologist for SAIC's Center for Advanced Information Technology. He has over 16 years of experience developing systems for the U.S. Navy and the Department of Defense working for SAIC. He holds a Masters degree in Software Engineering and a Bachelors degree in Computer Science.

Science Applications International Corporation

SAIC, a fortune 500 Company founded in 1969, is the largest employee-owned high-tech company in the U.S. with annual revenue in excess of $6 billion and over 43,000 employees with offices in more than 150 cities worldwide. Based in San Diego, SAIC offers a broad range of research and engineering expertise in technology development and analysis, computer system development and integration, technical support services, and computer software and services. SAIC works to solve complex technical problems in the areas of information technology, systems integration, telecommunications, national and international security, health systems and services, transportation, environmental systems and engineering, and space. Visit on the World Wide Web at: http://www.saic.com/

Disclaimer

Copyright

OPERATIONAL AGILITY

COMPOSING AND ORCHESTRATING MISSION CAPABILITY PACKAGES THROUGH BUSINESS PROCESS EXECUTION LANGUAGE (BPEL)

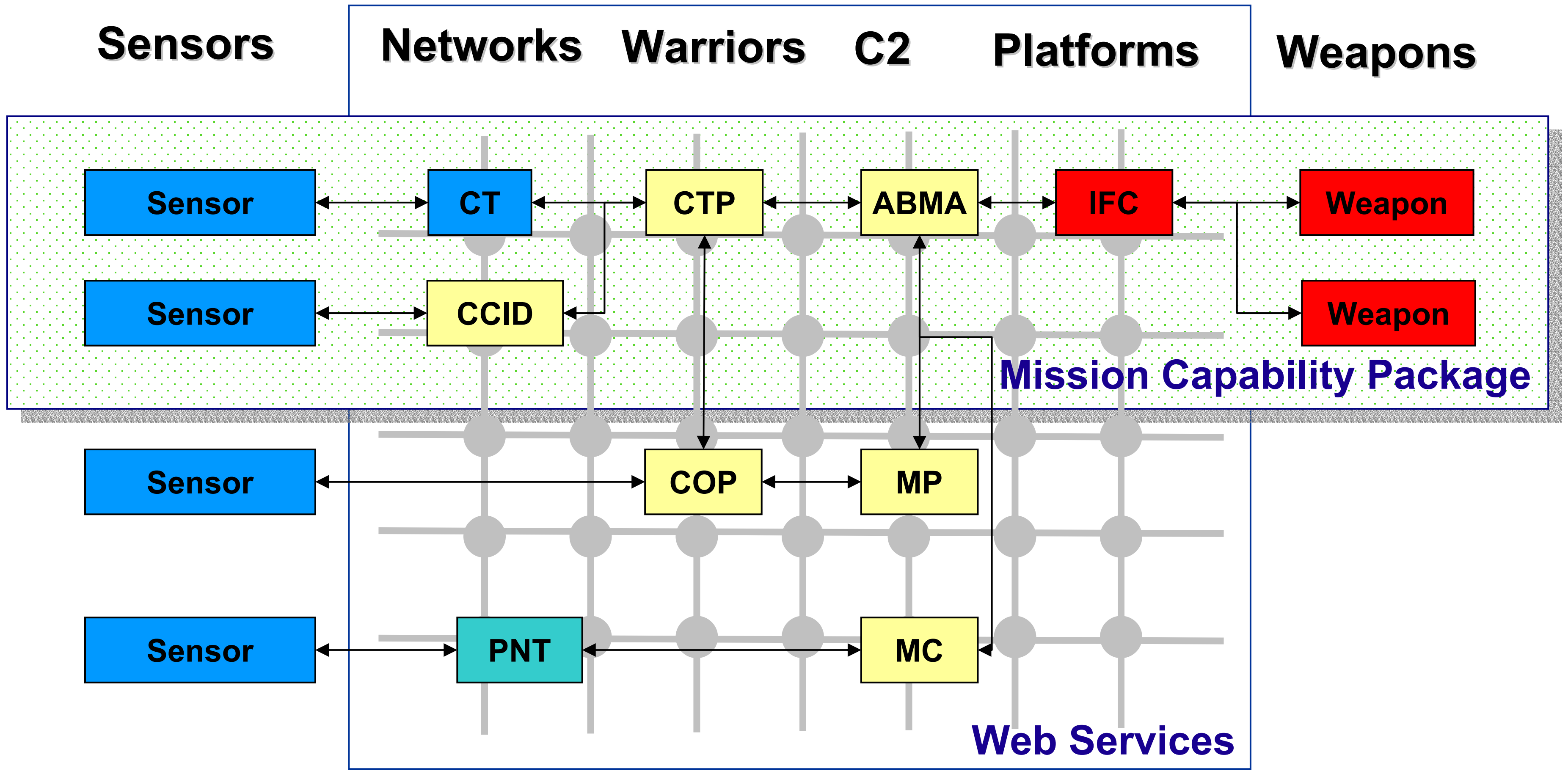
Gary R. Shaffer
Division Chief Technologist
Center for Advanced Information Technology, SAIC
(858) 826-5746
shafferg@saic.com

Mission Capability Package Defined…

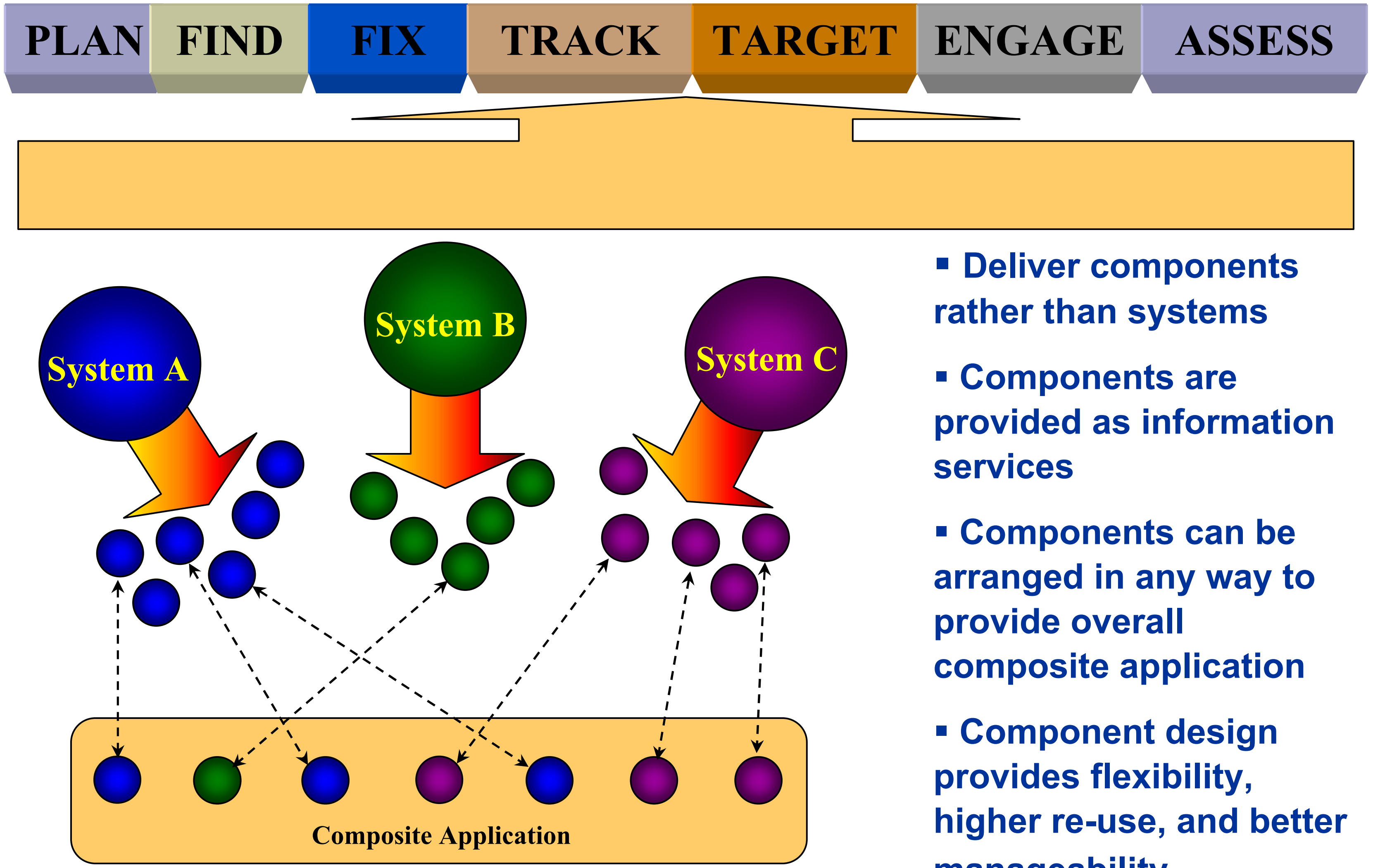
- Integrate a specific set of joint sensors, platforms, weapons, warriors, networks, command and control systems for the purpose of performing mission-specific engagements.
- Ability to dynamically re-configure and re-allocate assets "on the fly" based on current mission needs.

FnEP Masters Thesis, NPS, MAJ Robert Hesser and LCDR Dan Rieken

Mission Capability Packages



Technology Vision Applied

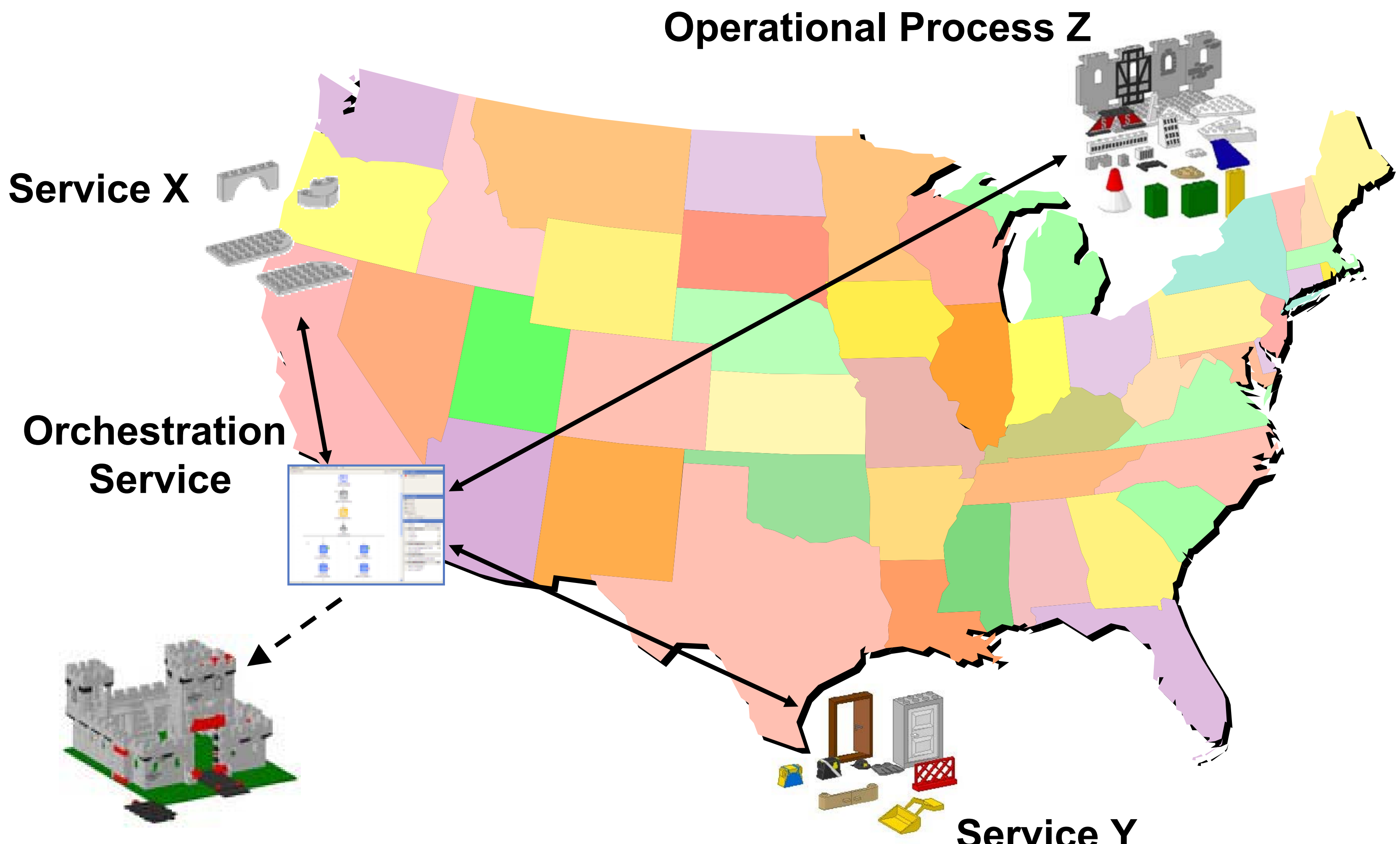


- Deliver components rather than systems
- Components are provided as information services
- Components can be arranged in any way to provide overall composite application
- Component design provides flexibility, higher re-use, and better manageability

Orchestration

- Generation of Mission Capability Packages (MCP) from deployed objects and/or services
- Composition can take place at design time or run time by Subject Matter Experts
- Binding takes place at run time (late binding)
  - ✓ Bindings can be based on
    - Specific end points
    - Selection of end points that met some specific criteria
- Dependent on key concepts/capabilities
  - ✓ Registration/Discovery
  - ✓ Identity Management
  - ✓ Web Services Definition Language
- Once deployed, newly composed services can be consumed by other services

Orchestrating Distributed Services



Mission Capability Package composed and orchestrated from services X, Y, & OP Z

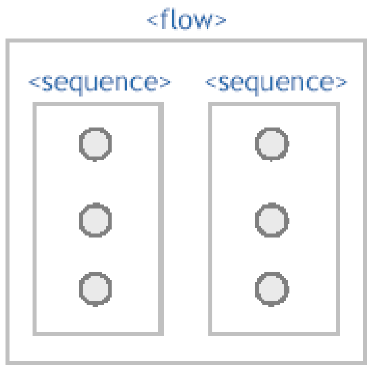
Orchestration for Web Services

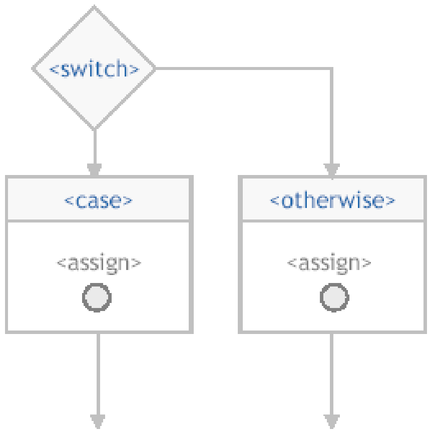
- Compose Operational Processes, Threads, FnEPs, MCPs, and/or ECMs from Business Process Execution Language (BPEL) for Web Services (BPEL4WS) based on standards work from
  - ✓ IBM's Web Services Flow Language (WSFL)
  - ✓ Microsoft's XLANG
- Specifies how collections of services are jointly used to realize more complex functionality
  - ✓ Describes the data shared between the services
  - ✓ Transactional states and joint exception handling
  - ✓ Separates the flow (execution) from the services themselves
  - ✓ Partnerships/Organizations
- Once deployed they can be consumed by other Operational Processes, MCPs, FnEPs, ECMs, and/or services

XML based Work flow for Web services….

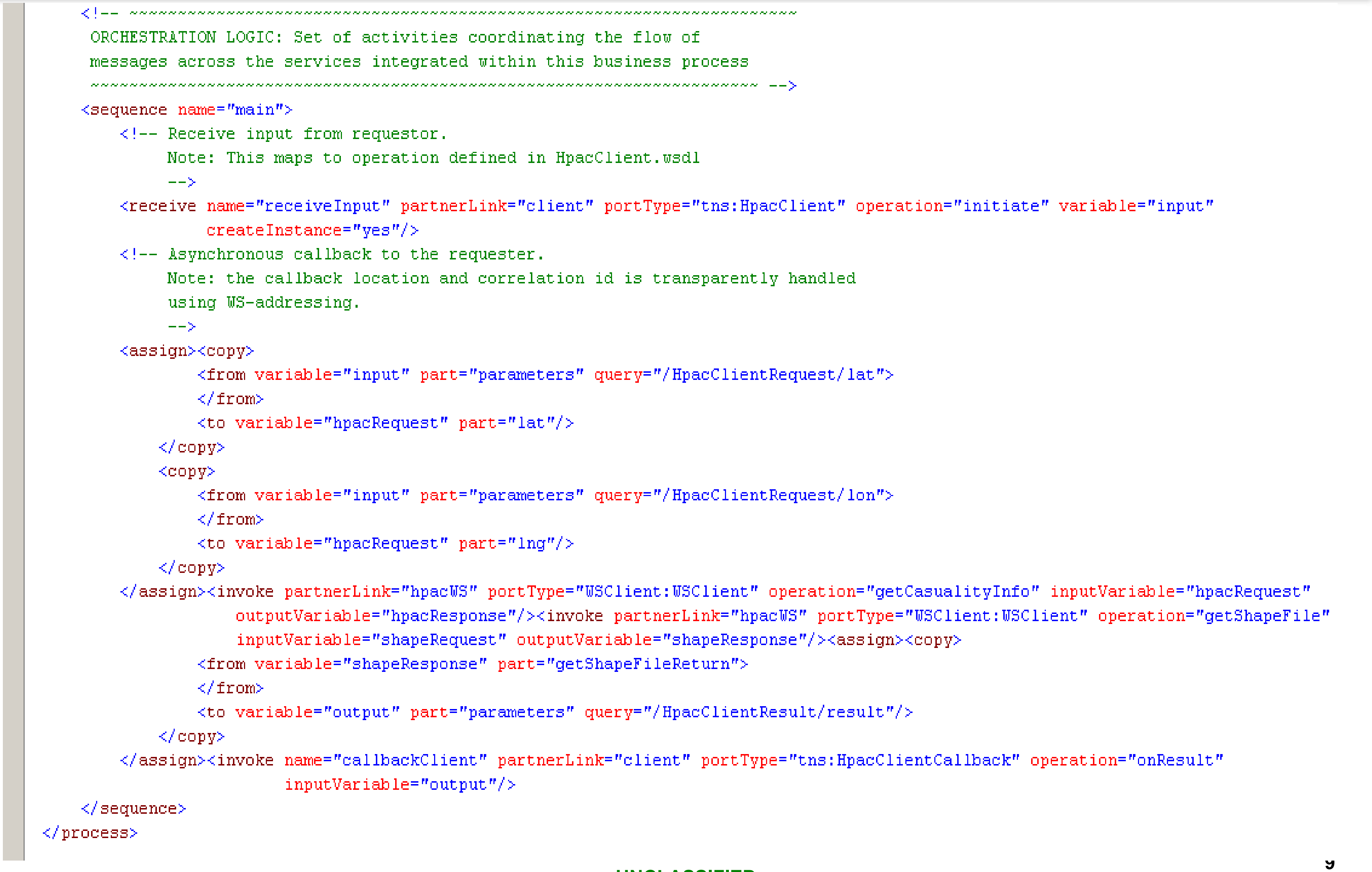
Its just XML…

- <invoke> a web service synchronously
- <assign> and manipulate XML documents
- <scope>, <faultHandlers> catch and manage exceptions
- Initiate asynchronous processing in parallel <flow> of execution
- <receive> asynchronous callbacks from long running services/processors
- <switch> on a set of pre-defined constraints



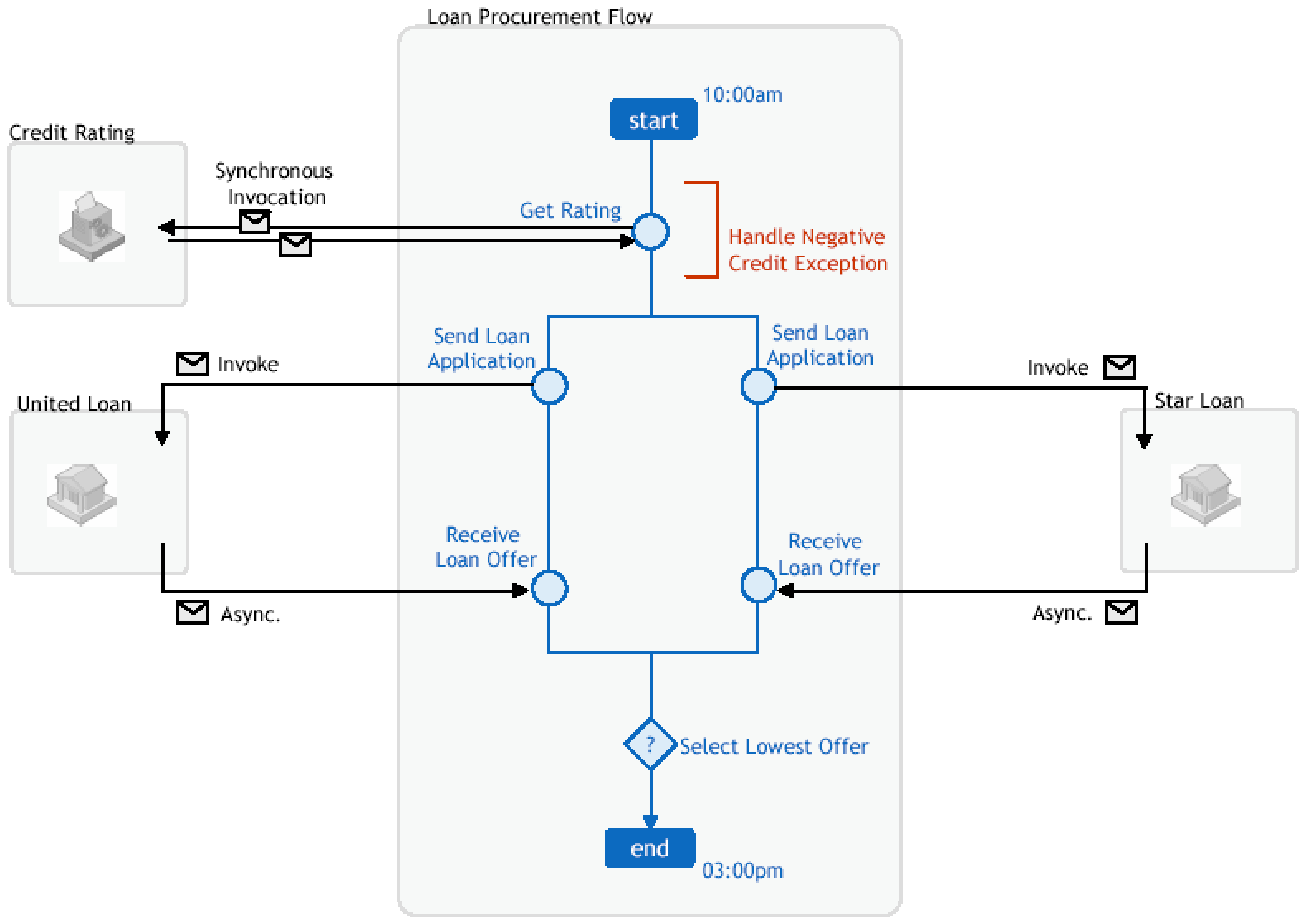


Example BPEL



```
    <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     ORCHESTRATION LOGIC: Set of activities coordinating the flow of
     messages across the services integrated within this business process
     ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
    <sequence name="main">
        <!-- Receive input from requestor.
             Note: This maps to operation defined in HpacClient.wsdl
             -->
        <receive name="receiveInput" partnerLink="client" portType="tns:HpacClient" operation="initiate" variable="input"
                 createInstance="yes"/>
        <!-- Asynchronous callback to the requester.
             Note: the callback location and correlation id is transparently handled
             using WS-addressing.
             -->
        <assign><copy>
                <from variable="input" part="parameters" query="/HpacClientRequest/lat">
                </from>
                <to variable="hpacRequest" part="lat"/>
            </copy>
            <copy>
                <from variable="input" part="parameters" query="/HpacClientRequest/lon">
                </from>
                <to variable="hpacRequest" part="lng"/>
            </copy>
        </assign><invoke partnerLink="hpacWS" portType="WSClient:WSClient" operation="getCasualityInfo" inputVariable="hpacRequest"
                    outputVariable="hpacResponse"/><invoke partnerLink="hpacWS" portType="WSClient:WSClient" operation="getShapeFile"
                    inputVariable="shapeRequest" outputVariable="shapeResponse"/><assign><copy>
                <from variable="shapeResponse" part="getShapeFileReturn">
                </from>
                <to variable="output" part="parameters" query="/HpacClientResult/result"/>
            </copy>
        </assign><invoke name="callbackClient" partnerLink="client" portType="tns:HpacClientCallback" operation="onResult"
                         inputVariable="output"/>
    </sequence>
</process>
```

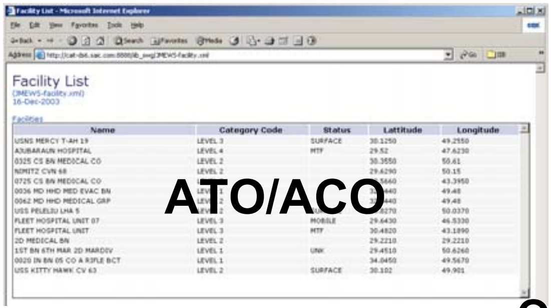
A BPEL Process Flow



An Employee-Owned Company

Time Critical Targeting Orchestration

From TBMCS (via SPF)



Facility List - Microsoft Internet Explorer

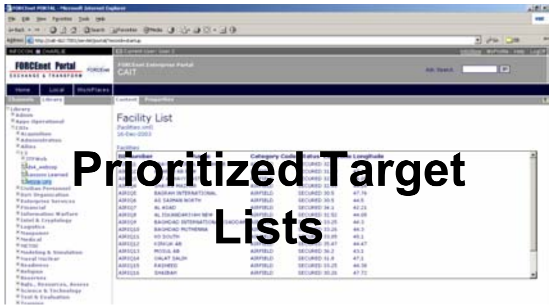
Facility List
(JMEWS-facility.xml)
16-Dec-2003

Facilities

Name	Category Code	Status	Lattitude	Longitude
USNS MERCY T-AH 19	LEVEL 3	SURFACE	30.1250	49.2550
AJUBARAUN HOSPITAL	LEVEL 4	MTF	29.52	47.6230
0325 CS BN MEDICAL CO	LEVEL 2		30.3550	50.61
NIMITZ CVN 68	LEVEL 2		29.6290	50.15
0725 CS BN MEDICAL CO	[illegible]		[illegible]	43.3950
0036 MD HHD MED EVAC BN	[illegible]		[illegible]	49.48
0062 MD HHD MEDICAL GRP	[illegible]		[illegible]	49.48
USS PELELIU LHA 5	[illegible]	[illegible]	[illegible]	50.0370
FLEET HOSPITAL UNIT 07	LEVEL 3	MOBILE	29.6430	46.5330
FLEET HOSPITAL UNIT	LEVEL 3	MTF	30.4820	43.1890
2D MEDICAL BN	LEVEL 2		29.2210	29.2210
1ST BN 6TH MAR 2D MARDIV	LEVEL 1	UNK	29.4510	50.6260
0020 IN BN 05 CO A RIFLE BCT	LEVEL 1		34.0450	49.5670
USS KITTY HAWK CV 63	LEVEL 2	SURFACE	30.102	49.901

ATO/ACO

From JIPTL and JTL

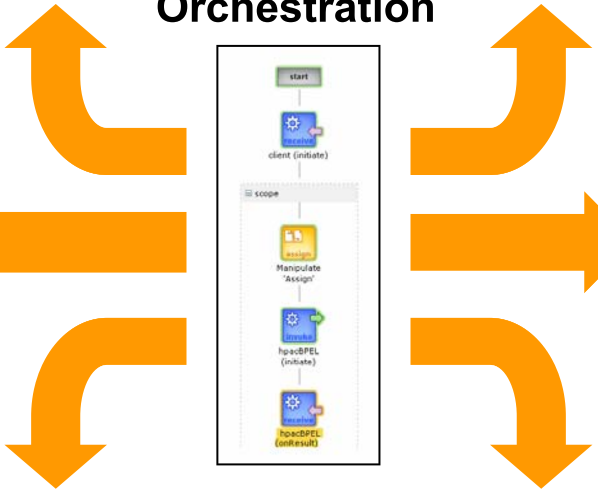


FORCEnet Portal

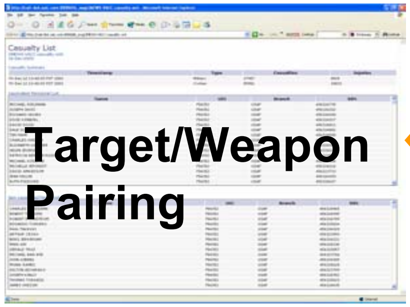
Facility List

Prioritized Target Lists

Orchestration



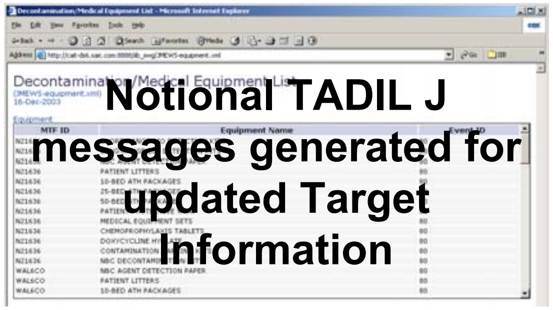
From JMEMS/WASP/JAWS



Casualty List

Target/Weapon Pairing

To ABM



Decontamination/Medical Equipment List - Microsoft Internet Explorer

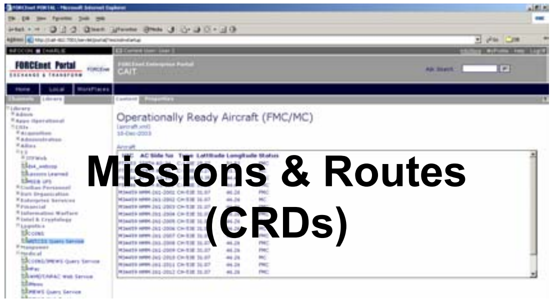
Decontamination/Medical Equipment List
(JMEWS-equipment.xml)
16-Dec-2003

Equipment

MTF ID	Equipment Name	Event ID
[illegible]	[illegible]	80
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	80
N21636	PATIENT LITTERS	80
N21636	10-BED ATH PACKAGES	80
N21636	[illegible]	80
N21636	[illegible]	[illegible]
N21636	[illegible]	[illegible]
N21636	MEDICAL EQUIPMENT SETS	80
N21636	CHEMOPROPHYLAXIS TABLETS	80
N21636	[illegible]	80
N21636	[illegible]	80
N21636	[illegible]	80
WAL6CO	NBC AGENT DETECTION PAPER	80
WAL6CO	PATIENT LITTERS	80
WAL6CO	10-BED ATH PACKAGES	80

Notional TADIL J messages generated for updated Target Information

From SPF

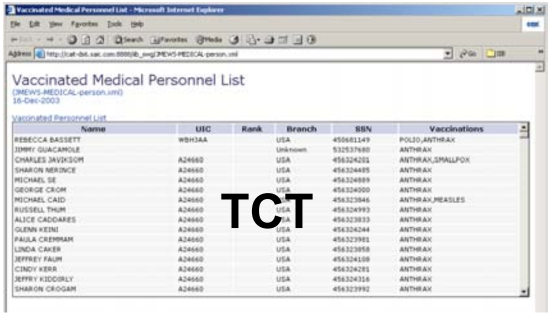


FORCEnet Portal

Operationally Ready Aircraft (FMC/MC)

Missions & Routes (CRDs)

From WebCOP



Vaccinated Medical Personnel List - Microsoft Internet Explorer

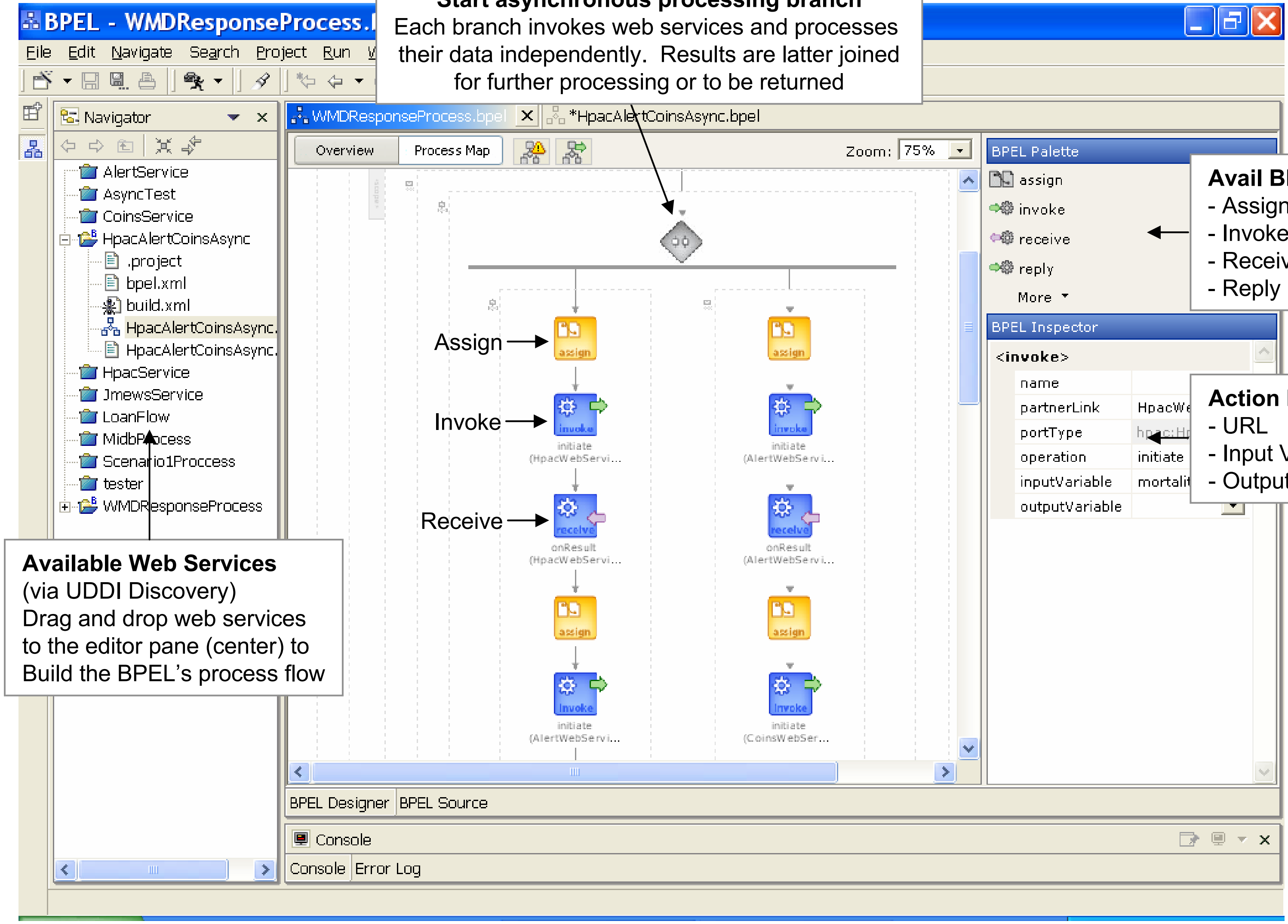
Vaccinated Medical Personnel List
(JMEWS-MEDICAL-person.xml)
16-Dec-2003

Vaccinated Personnel List

Name	UIC	Rank	Branch	SSN	Vaccinations
REBECCA BASSETT	WBH3AA		USA	450681149	POLIO,ANTHRAX
JIMMY GUACAMOLE			Unknown	532537680	ANTHRAX
CHARLES JAVIKSOM	A24660		USA	456324201	ANTHRAX,SMALLPOX
SHARON NERINCE	A24660		USA	456324485	ANTHRAX
MICHAEL SE	A24660		USA	456324889	ANTHRAX
GEORGE CROM	A24660		USA	456324000	ANTHRAX
MICHAEL CAID	A24660		[illegible]	456323846	ANTHRAX,MEASLES
RUSSELL THUM	A24660		USA	456324993	ANTHRAX
ALICE CADDARES	A24660		[illegible]	456323833	ANTHRAX
GLENN KEINI	A24660		USA	456324244	ANTHRAX
PAULA CREMMAM	A24660		USA	456323981	ANTHRAX
LINDA CAKER	A24660		USA	456323858	ANTHRAX
JEFFREY FAUM	A24660		USA	456324108	ANTHRAX
CINDY KERR	A24660		USA	456324281	ANTHRAX
JEFFRY KIDDIRLY	A24660		USA	456324316	ANTHRAX
SHARON CROGAM	A24660		USA	456323992	ANTHRAX

TCT

Building Operational Processes



Start asynchronous processing branch
Each branch invokes web services and processes their data independently. Results are latter joined for further processing or to be returned

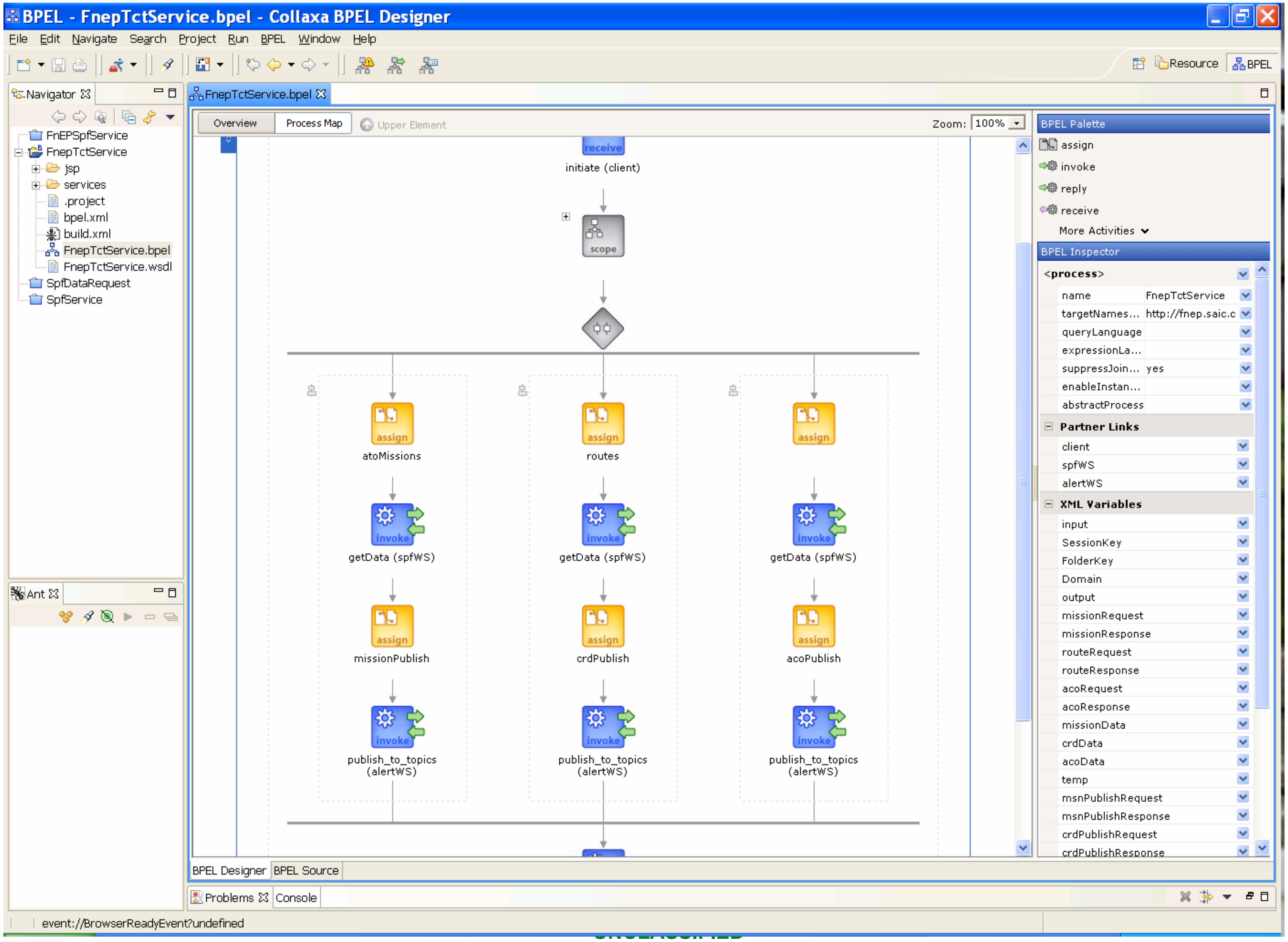
Avail BPEL Actions
- Assign
- Invoke
- Receive
- Reply

Action Properties
- URL
- Input Variables
- Output Variables

Available Web Services
(via UDDI Discovery)
Drag and drop web services to the editor pane (center) to Build the BPEL's process flow

An Employee-Owned Company

TCT BPEL (Design Time)



<process>

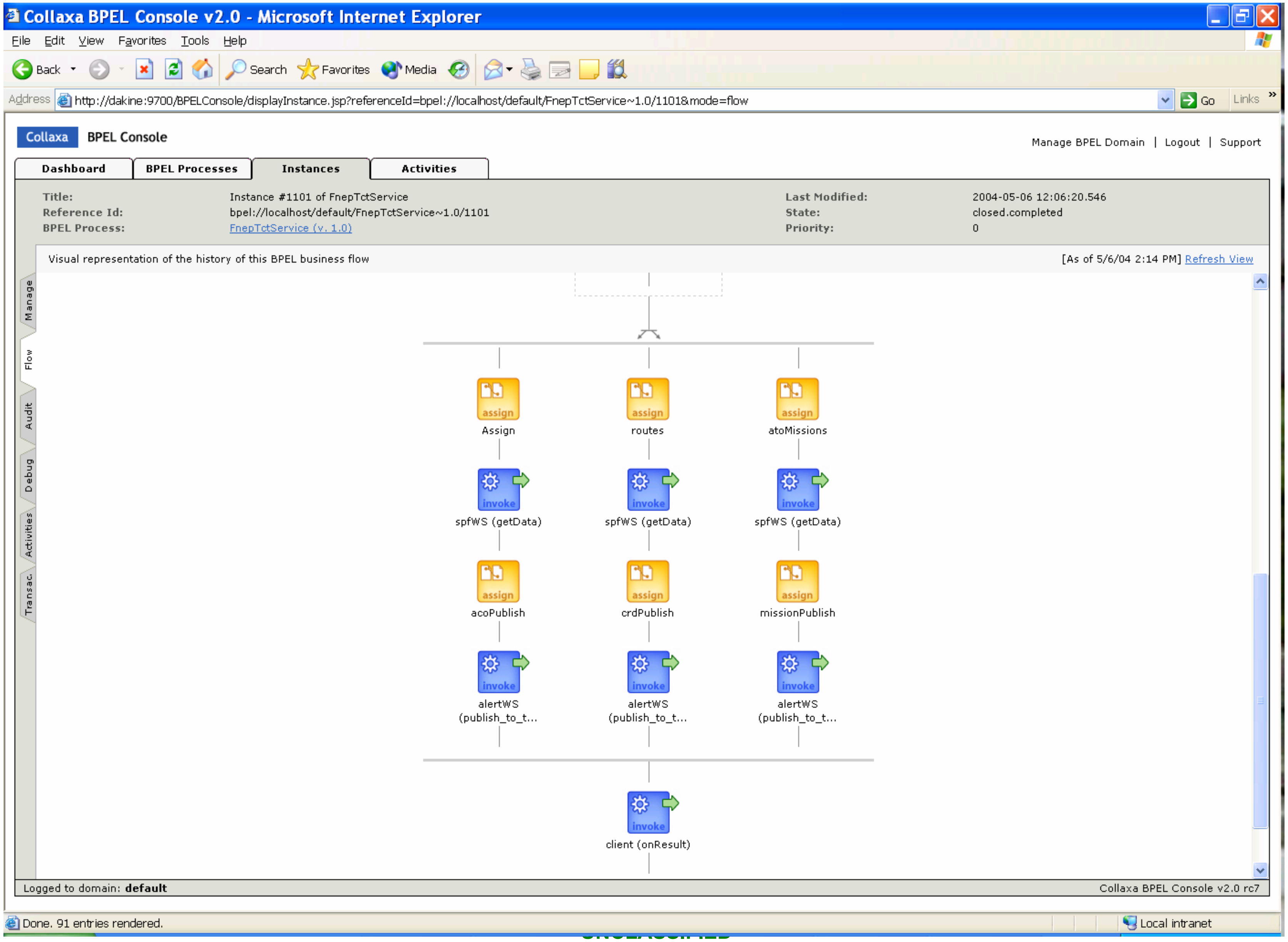
name	FnepTctService
targetNames...	http://fnep.saic.c
queryLanguage	
expressionLa...	
suppressJoin...	yes
enableInstan...	
abstractProcess	

Partner Links
- client
- spfWS
- alertWS

XML Variables
- input
- SessionKey
- FolderKey
- Domain
- output
- missionRequest
- missionResponse
- routeRequest
- routeResponse
- acoRequest
- acoResponse
- missionData
- crdData
- acoData
- temp
- msnPublishRequest
- msnPublishResponse
- crdPublishRequest
- crdPublishResponse

event://BrowserReadyEvent?undefined

TCT BPEL (Execution Time)



Collaxa BPEL Console v2.0 - Microsoft Internet Explorer

Address http://dakine:9700/BPELConsole/displayInstance.jsp?referenceId=bpel://localhost/default/FnepTctService~1.0/1101&mode=flow

Title:	Instance #1101 of FnepTctService	Last Modified:	2004-05-06 12:06:20.546
Reference Id:	bpel://localhost/default/FnepTctService~1.0/1101	State:	closed.completed
BPEL Process:	FnepTctService (v. 1.0)	Priority:	0

Visual representation of the history of this BPEL business flow

[As of 5/6/04 2:14 PM] Refresh View

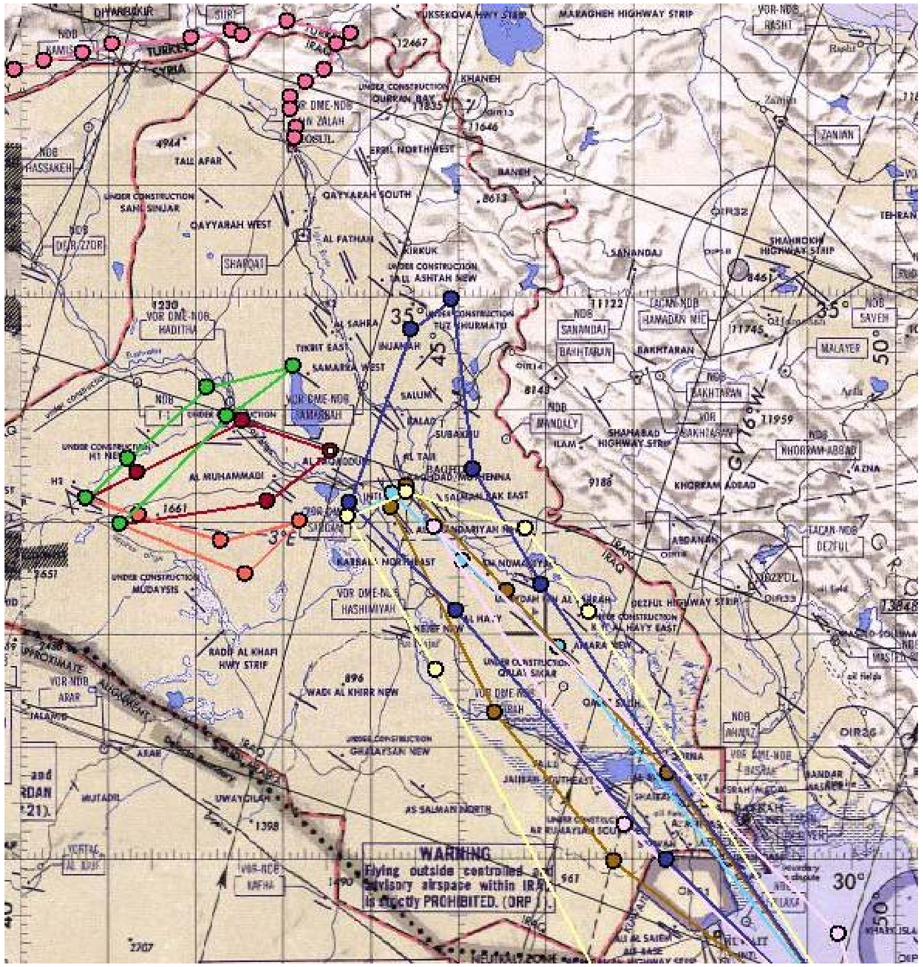
Logged to domain: default

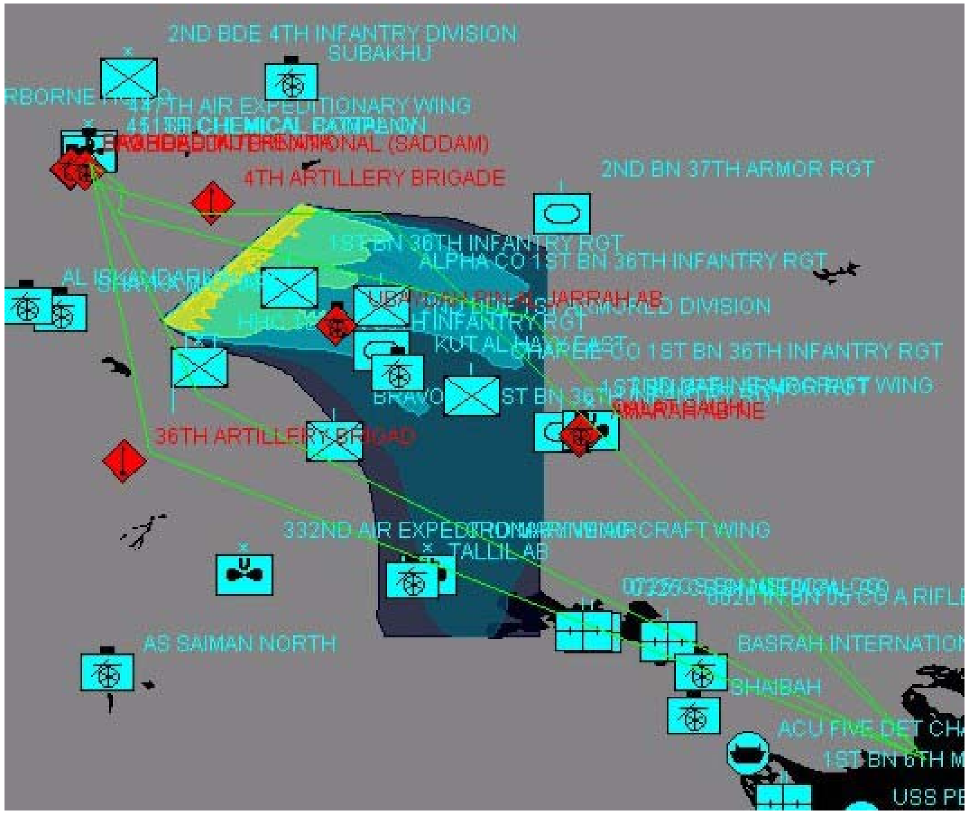
Collaxa BPEL Console v2.0 rc7

Done. 91 entries rendered.

Local intranet

TCT Filtering using BPEL

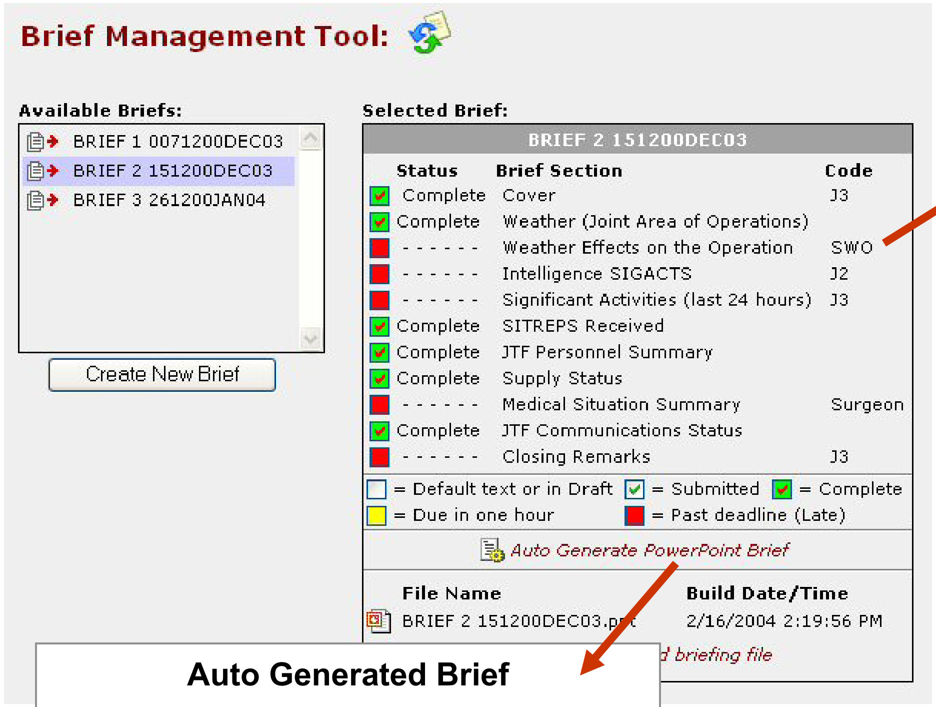




- Time
- Time On Target
- Weapon/Target Pairing
- Mission Type
- Proximity

An Employee-Owned Company

Web-Based JTF Staff Brief Builder



Brief Management Tool:

Available Briefs:

- BRIEF 1 0071200DEC03
- BRIEF 2 151200DEC03
- BRIEF 3 261200JAN04

Create New Brief

Selected Brief:

BRIEF 2 151200DEC03

Status	Brief Section	Code
✓ Complete	Cover	J3
✓ Complete	Weather (Joint Area of Operations)	
🟥 - - - - - -	Weather Effects on the Operation	SWO
🟥 - - - - - -	Intelligence SIGACTS	J2
🟥 - - - - - -	Significant Activities (last 24 hours)	J3
✓ Complete	SITREPS Received	
✓ Complete	JTF Personnel Summary	
✓ Complete	Supply Status	
🟥 - - - - - -	Medical Situation Summary	Surgeon
✓ Complete	JTF Communications Status	
🟥 - - - - - -	Closing Remarks	J3

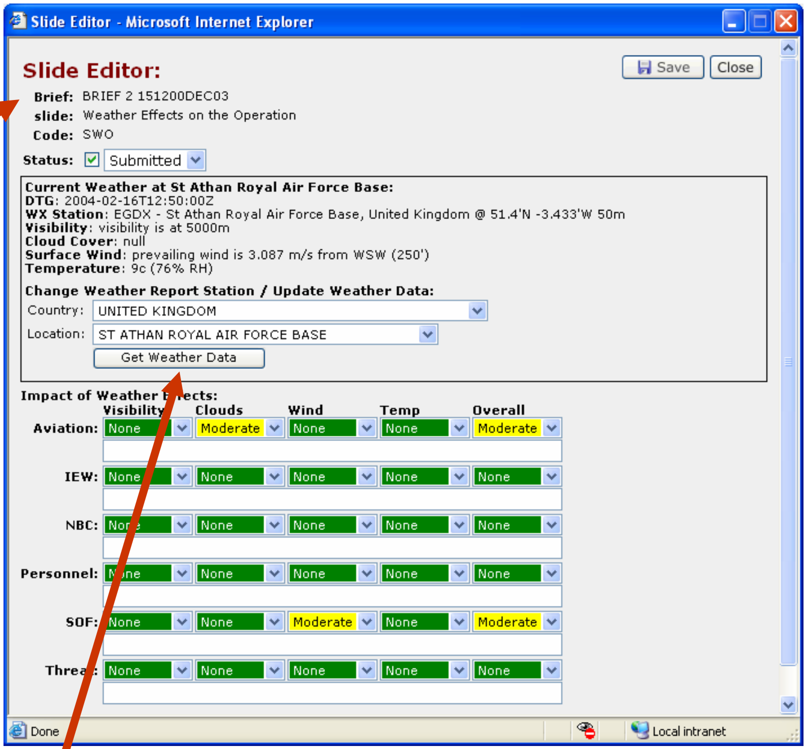
☐ = Default text or in Draft ☑ = Submitted ✓ = Complete
🟨 = Due in one hour 🟥 = Past deadline (Late)

Auto Generate PowerPoint Brief

File Name	Build Date/Time
BRIEF 2 151200DEC03.ppt	2/16/2004 2:19:56 PM

d briefing file

Weather Effects on Operations Slide



Slide Editor - Microsoft Internet Explorer

Slide Editor: Save Close

Brief: BRIEF 2 151200DEC03
slide: Weather Effects on the Operation
Code: SWO

Status: ☑ Submitted

Current Weather at St Athan Royal Air Force Base:
DTG: 2004-02-16T12:50:00Z
WX Station: EGDX - St Athan Royal Air Force Base, United Kingdom @ 51.4'N -3.433'W 50m
Visibility: visibility is at 5000m
Cloud Cover: null
Surface Wind: prevailing wind is 3.087 m/s from WSW (250')
Temperature: 9c (76% RH)

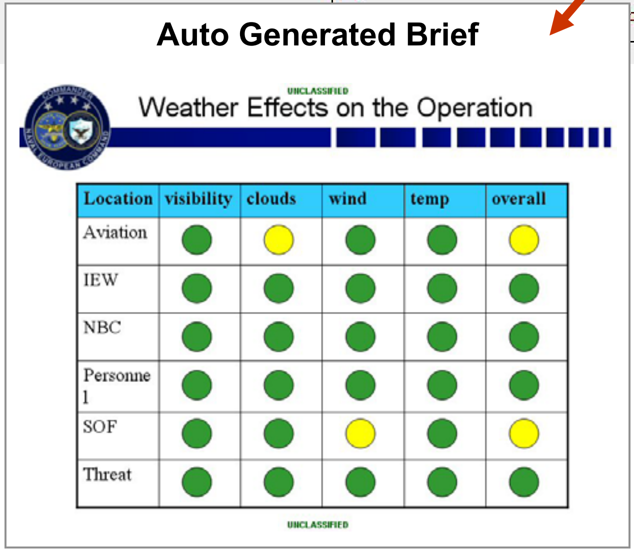
Change Weather Report Station / Update Weather Data:
Country: UNITED KINGDOM
Location: ST ATHAN ROYAL AIR FORCE BASE
Get Weather Data

Impact of Weather Effects:

	Visibility	Clouds	Wind	Temp	Overall
Aviation:	None	Moderate	None	None	Moderate
IEW:	None	None	None	None	None
NBC:	None	None	None	None	None
Personnel:	None	None	None	None	None
SOF:	None	None	Moderate	None	Moderate
Threat:	None	None	None	None	None

Done Local intranet

Auto Generated Brief



UNCLASSIFIED

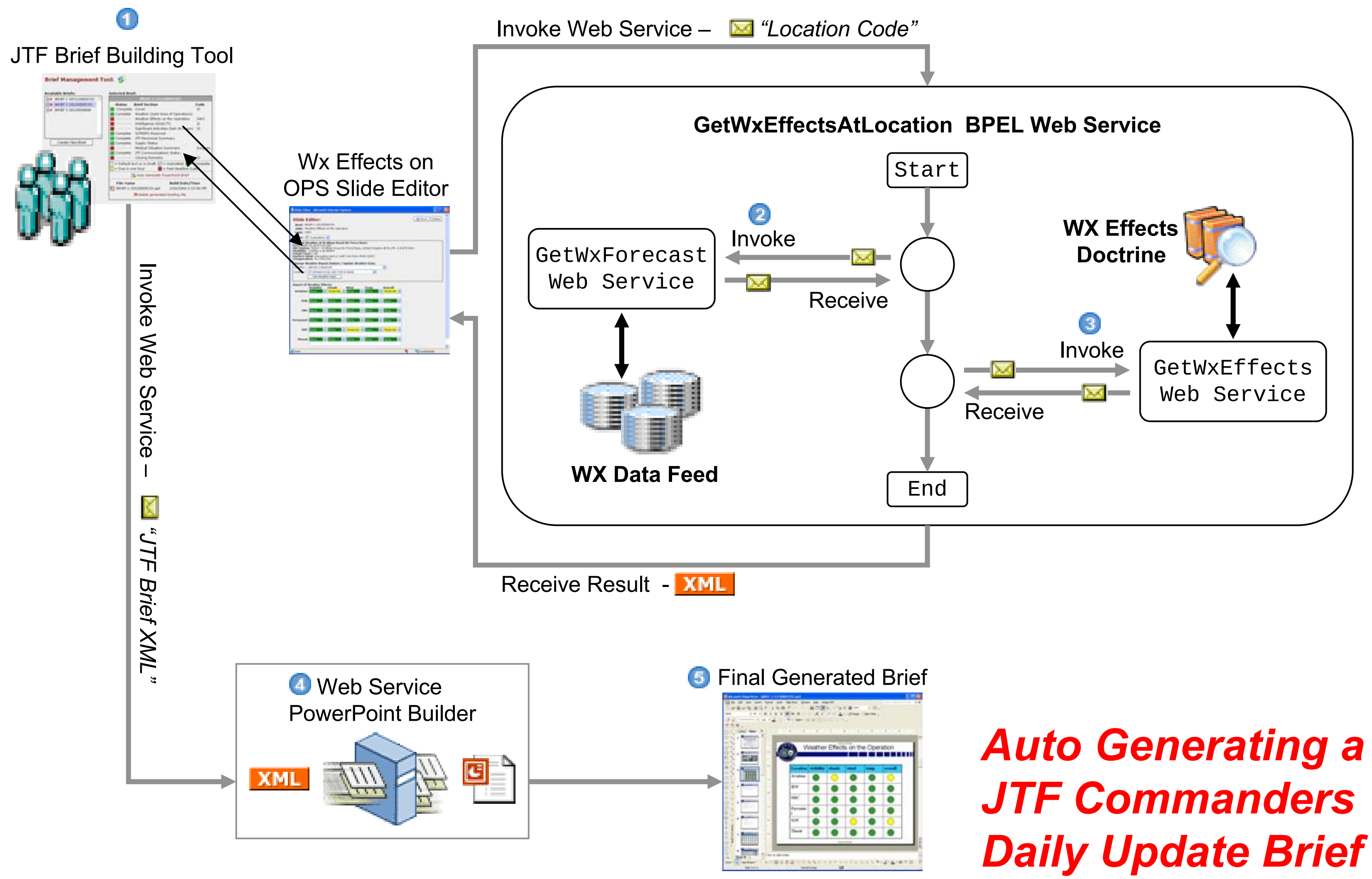
Weather Effects on the Operation

Location	visibility	clouds	wind	temp	overall
Aviation	🟢	🟡	🟢	🟢	🟡
IEW	🟢	🟢	🟢	🟢	🟢
NBC	🟢	🟢	🟢	🟢	🟢
Personnel	🟢	🟢	🟢	🟢	🟢
SOF	🟢	🟢	🟡	🟢	🟡
Threat	🟢	🟢	🟢	🟢	🟢

UNCLASSIFIED

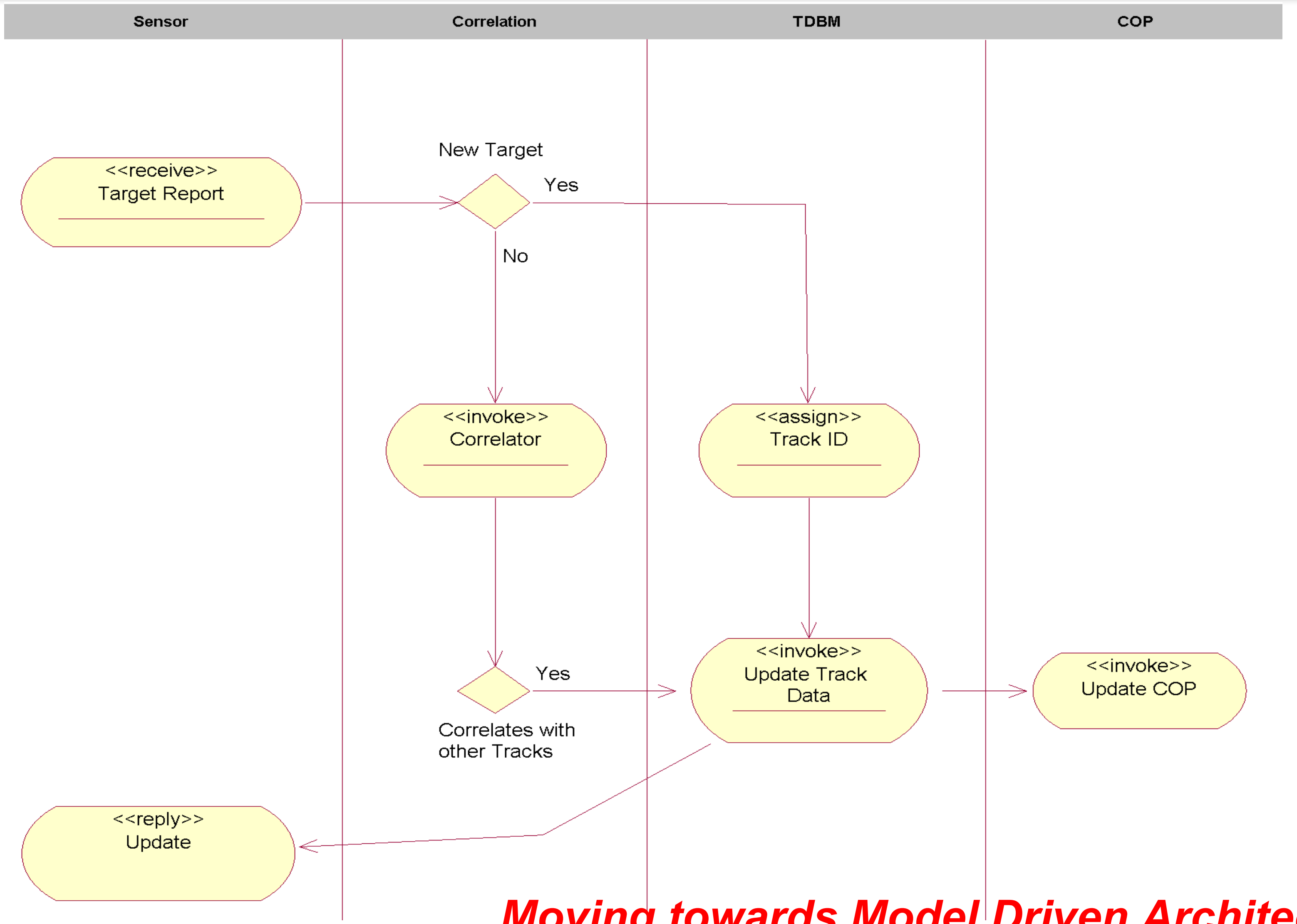
Invokes Operational Process that gathers data and processes it based on "FM 34-81-1 Weather effect on Operations Field Manual"

Operational Process Flow



Auto Generating a JTF Commanders Daily Update Brief

Modeling Operational Processes in UML



Moving towards Model Driven Architectures

An Employee-Owned Company

Questions

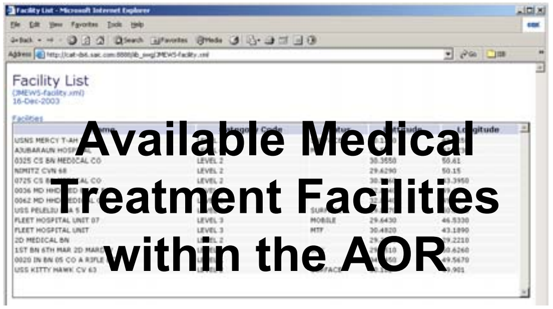
Acronyms/Terms/Definitions

MCP	Mission Capability Package
FnEP	FORCEnet Engagement Pack
ECM	Evaluation Capability Module
BPEL	Business Process Execution Language
BPEL4WS	BPEL for Web Services
WSDL	Web Services Defintion Language
TCT	Time Critical Targeting
IFC	Integrated Fire Control
ABMA	Automated Battle Management Aid
CT	Composite Tracking
CCID	Composite Combat ID
COP	Common Operational Picture
CTP	Common Tactical Picture
C2	Command & Control
UML	Unified Modeling Language
JTF	Joint Task Force

SAIC An Employee-Owned Company

Contingency Planning Orchestration

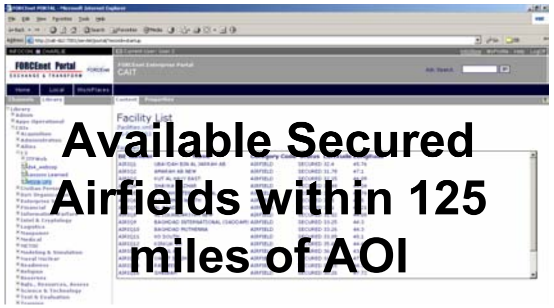
From JMeWS



Facility List
JMEWS-facility.xml
16-Dec-2003

Available Medical Treatment Facilities within the AOR

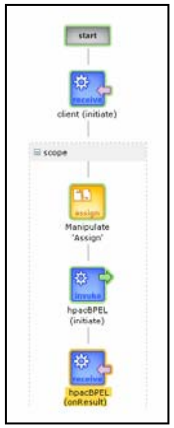
From MIDB



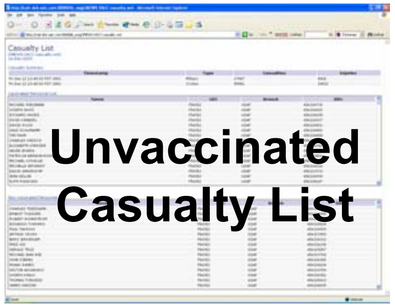
FORCEnet Portal
Facility List

Available Secured Airfields within 125 miles of AOI

Orchestration



From HPAC/JMEWS/COINS



Casualty List

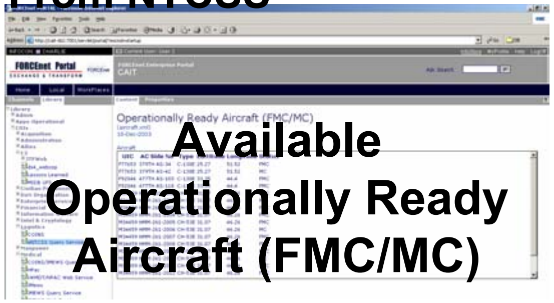
Unvaccinated Casualty List

From JMeWS

Decontamination/Medical Equipment List
JMEWS-equipment.xml

Available Decontamination and Medical Supplies within the AOR

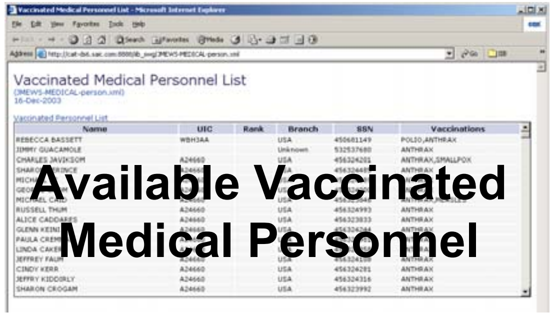
From NTCSS



FORCEnet Portal
Operationally Ready Aircraft (FMC/MC)

Available Operationally Ready Aircraft (FMC/MC)

From COINS & JMeWS



Vaccinated Medical Personnel List
JMEWS-MEDICAL-person.xml
16-Dec-2003

Available Vaccinated Medical Personnel