Dynamic Service Integration for Reliable and Sustainable Capability Provision

Lu Liu[a], Jie Xu[b], Duncan Russell[c], John K Davies[d], David Webster[b], Zongyang Luo[e] & Colin Venters[f]

[a] *School of Engineering and Information Sciences, Middlesex University, London, UK*
[b] *School of Computing, University of Leeds, Leeds, UK*
[c] *Image Analysis Ltd, Saltaire, UK*
[d] *BAE Systems Integrated Systems Technologies, Victory Point, Frimley, UK*
[e] *Jinton Group, Changzhou, Jiangsu, P.R.China*
[f] *Kyungpook National University, South Korea*

(Received 15 June 2009; final version received 12 March 2010)

The move towards Network Enabled Capability (NEC) by the UK Ministry of Defence is designed to achieve enhanced military effect through the networking and coherent integration of existing and future resources including sensors, weapon systems, and decision makers to achieve a more flexible and responsive military. This paper addresses the existing reliability and sustainability issues of large-scale military systems and proposes new architectural approaches of dynamic service integration for NEC to adapt to evolution occurring in services and capability for constructing next generation software-intensive military systems. The reliability and performance of the proposed architectural approaches have been verified through modelling and simulation of Service Oriented Architecture for NEC and demonstrated through developing and testing a NEC system for a region surveillance capability scenario. The experimental results indicate that the proposed architectural approaches provide a high-level of reliability and sustainability in the provision of NEC.

Keywords: Service-Oriented Architecture, Network-Enabled Capability, Service Integration

1. Introduction

Service-Oriented Architecture (SOA) is concerned with the structure of service provision and consumption and the infrastructure to support the interactions. The architecture is made of service suppliers and consumers, with suppliers advertising through registries or brokers for consumers to discover (Russell and Xu, 2007, Alonso et al., 2004). Loose coupling is one of the key architectural principles of SOA. This enables services to maintain a relationship that minimises dependencies and only requires maintaining an awareness of each other. The loose coupling of SOA enables service implementations to be inter-changed and modified.

The use of SOA has been motivated by many industries changing focus from product delivery to service-based delivery. The focus on service delivery has also been apparent in software, where networking has become faster, more reliable and more available through reduced cost. The approach to SOA in software enables business process integration that characterises business functions as services, and integrates dynamically across departments and organisations. The conceptual SOA can be used to integrate businesses, systems and computing at runtime (Tsai et al., 2006a) by using different levels of abstraction.

Capability is the ability to achieve a specific wartime objective (DoD, 1994). Network Enabled Capability (NEC) is the U.K. Ministry of Defence (MoD)'s response to the quickly changing conflict environment in which its forces must operate. Using the definition adopted in the Network Enabled Capability Through Innovative System Engineering (NECTISE) project, NEC is the integration of assets to fulfil a mission objective (Liu et al., 2008a). The NEC initiative recognises that offering functionality is the main requirement in supporting military capability, and that functionality can be delivered without ownership of the delivery mechanism. From the Defence Industrial Strategy: "We are seeing a shift away from platform oriented programme towards a capability-based approach"(UK Ministry of Defence, 2005a), suppliers can be allowed to respond to customers needs, providing the delivery of appropriate and up-to-date solutions into the military rather than responding to requirements for specific equipments.

To respond to this need, the U.K. EPSRC and BAE Systems jointly funded the NECTISE project, which is addressing the question of how industries deliver elements that contribute to NEC for its customers, taking account of the aims summarised in the

2005 Defence Industrial Strategy (UK Ministry of Defence, 2005a). The architecture for NEC is about integrating distributed systems and networks by addressing such concerns as availability, accessibility, integrity, reliability, security, maintainability and resilience. One of the objectives of the NECTISE project is to develop a systematic approach that would lead to evolutionary architectures for through-life evolution, which is addressed in this paper.

NEC is about the coherent integration of sensors, decision-makers, weapon systems and support capabilities to achieve the desired effect (UK Ministry of Defence, 2005b). However, ongoing evolution, such as evolution of services and evolution of requirements, significantly influence the reliability of NEC provision (Liu et al., 2008a, Liu et al., 2009). The reliability means continuity of correct service (Avizienis et al., 2004). In order to provide reliable and sustainable capability in the new context of NEC, new approaches are needed to cope with ongoing evolution in dynamic environments without halting the operation of the NEC system.

In this paper, we present an innovative model to cope with the effect of the evolution of services and requirements for the provision of reliable and sustainable military capability in a network enabled environment. The main contributions of our work are: (1) using the concepts of evolutionary service-oriented architecture (Liu et al., 2008b) and dynamic workflow management to enable dynamic service integration for provision of reliable and sustainable military capability; (2) a SOA-based approach of the mechanisms of redundant service binding and dynamic service discovery to cope with evolution of services in the provision of capability; and (3) a self-adaptive approach that is able to dynamically evaluate the influence of evolution of capability and self-configure services

to adapt to the evolution of capability. The reliability of implementing these approaches for delivery of capability has been evaluated by simulations in a dynamic environment. The architectural approaches have been used to develop a demonstration system for a region surveillance capability scenario.

The rest of the paper is organised as follows. Section 2 discusses related work on Web service composition and integration. The service-oriented architecture for NEC is discussed in Section 3. In Section 4, the reliability and performance of the architecture for the provision of NEC are evaluated by simulations in a dynamic environment. The NEC demonstration system for regional surveillance to illustrate the use of SOA for NEC is introduced in Section 5. In Section 6, conclusions are drawn and future work is described.

2. Related Work

Web services implement SOA. In the last decade, a number of Web Service composition frameworks and applications have been developed. Alonso et.al (Alonso et al., 2004) described six different dimensions of web service composition models which can make different assumption of types of components considered. The disadvantages of composition models are making composition work more involved because of the heterogeneity of the components. Web Services Composite Application Framework (WS-CAF) (OASIS, 2006) is an open an open framework developed by OASIS. The purpose of the OASIS WS-CAF is to define a generic and open framework for applications that contain multiple services used in combination.

eFlow (Casati et al., 2000), developed by HP, is a system for the specification, enactment and management of composite services (Casati and Sha, 2001). Composite services are modelled by a graph which defines the flow of service invocations. eFlow

provides the dynamic features to cope with the rapidly evolving business environment where Web services are used. BPEL (Andrews et al., 2003) is a standard business process execution language which forms the necessary technical foundation for multiple usage patterns including both the descriptions of process interface for business protocols and executable process models. BPEL is based on BPEL4WS submission from Microsoft, IBM and BEA. However, vendor specific BPEL engines do not all support the complete standard and many of them have their own proprietor extensions. In addition, BPEL does not support quality of service and security, which make it difficult to capture real-time execution requirements. In the SOA for NEC, the definition of the term 'service' is not limited to Web Services and is not restricted to specific technologies. Services include other system resources and processes, which are addressed in this paper.

3. Design for Evolution

Architectural design for evolution is one of objectives set in the NECTISE project. A systematic approach needs to be developed that would lead to flexible architectures for through-life evolution. A process architecture for the agile delivery of capability that enables evolution is illustrated in Figure 1.

The architecture for agile delivery of capability can be classified into three layers: capability layer, integration layer and service layer. In the capability layer, new capability requirements are determined through long to medium term capability planning. In the integration layer, configurations and specifications of capability are defined based on these requirements. Configuration defines the actual combination of services used to implement the capability. This allows the abstract concept of the capability to be defined in terms of a set of abstract specifications which cover service interfaces, functional and

non-functional behaviours. Functional behaviour includes describing data formats, pre and post conditions and the operation performed by the service. Non-functional behaviour includes accuracy, security, timing and other quality of service parameters. Capability in terms of quality of services is out of the scope of the current work.

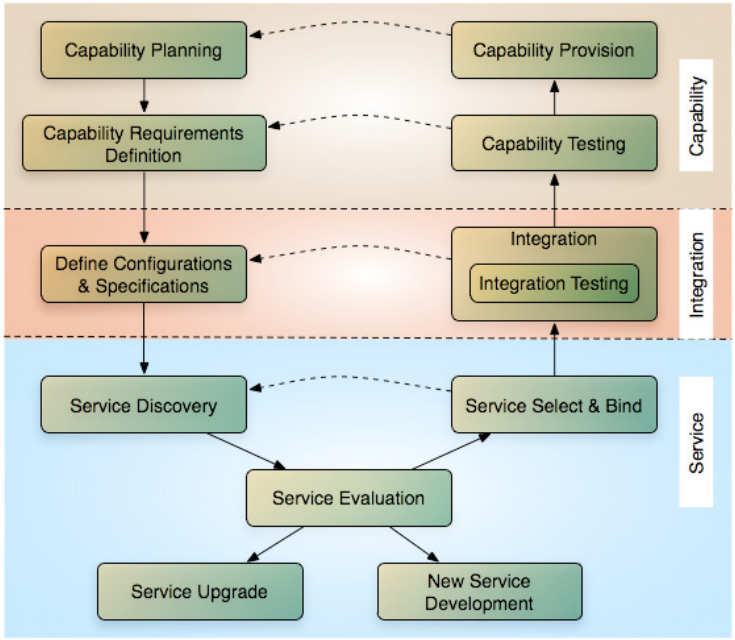


Figure 1: Agile delivery of capability

In the service layer, services which may be provided in the military networked systems and platforms are evaluated based on cost of service and quality of service. The gap between current services and requirements of specifications are identified. The service evaluation function decides whether to develop new services or bind to existing services for the provision of military capability, or both. Finally, the selected services are integrated with a dynamic workflow and tested (Tsai et al., 2008) in order to deliver a high-level military capability. The decision can be made according to the timescale of provision of capability. In response to urgent requests for change of capability, the

evaluation function prefers to dynamically bind to (or upgrade) current service rather than to develop new service from scratch.

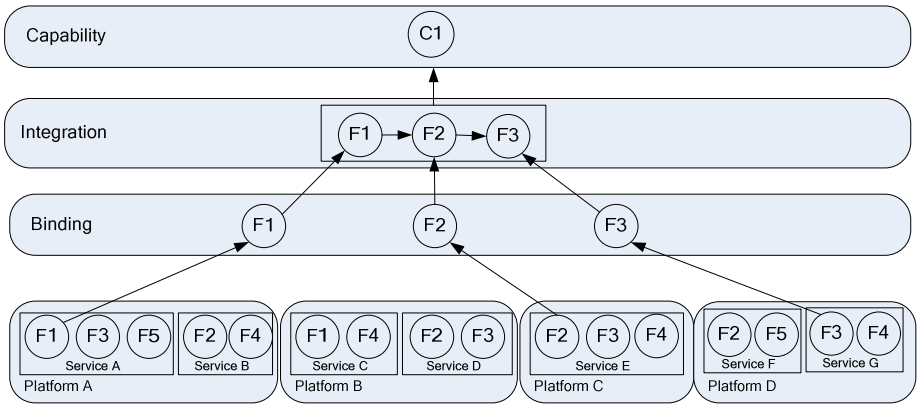


Figure 2: Service-oriented architecture for the delivery of capability

Figure 2 illustrates SOA for the delivery of capability where only binding to existing services is considered. In this architecture, each platform provides a number of services, each service performs a set of functions, and these can be integrated to form a higher level of functionality to deliver a capability. Dynamic binding allows common functions to be identified in different system implementations across platforms. The architecture enables functions from different systems across platforms to be integrated to provide capability in a loosely coupled manner. For example, a tank is a platform which provides services, such as movement, surveillance, and weapon delivery services. The surveillance service includes functions for environmental surveillance, situation surveillance and target surveillance. The environmental surveillance function may be combined with other functions to form a higher level metrological service that contributes to an Airborne Strike capability.

3.1 Motivations for Evolution

Using the definition adopted in NECTISE, NEC is the integration of systems to fulfil a mission objective. NEC requires system integration of independent services that can evolve, operate in a dependable manner where evolution of services can be handled without interruption of the provision of NEC (Russell et al., May 2008).

For provision of reliable and sustainable capability, evolution must be coped with in dynamic environments. The motivations for evolution for NEC are multiple (Webster et al., 2008a), such as

- Fault removal
- Customer Need
- Competition – between suppliers in provision and the enemy in operations
- Technology Development and Change (e.g. initial cost, maturation, phase-out)
- Standards (technical, etc.)
- Efficiency
- Architectural optimization
- Obsolescence
- Legislation/Litigation

Evolution occurring in capability is usually caused by one or multiple drivers mentioned above. For example, improvements in capability required to meet emerging operational threats may also take advantage of advances in technology. In SOA, services can have their own lifecycles independent of the lifecycle for capability. Evolution could occur either in services or in capability, or both. This can lead to compatibility issues and affect the reliability of the provision of capability. In order to provide reliable and sustainable capability, upgrades needs to be performed without having to lose capability

by taking equipment out of service for prolonged periods. In the next subsection, two new architectural mechanisms – redundant service binding and dynamic service discovery - are presented which are able to adapt to different types of evolution of services at runtime.

3.2 Evolution of Services

Ongoing changes, such as changes of platforms (e.g., adding and removing services from platforms) and changes of networks (e.g., network nodes joining and leaving the network), can influence the dependability of capability provision. From a service perspective, evolution of services may cause problems for the provision of reliable and sustainable capability which is capable of coping with changes without halting the provision of NEC. When a service is updated, it should be ensured that it conforms to the requirement in the integration layer. The provision of capability should not be interrupted even if one of the bound services offering the requested functions is replaced or updated from the platform.

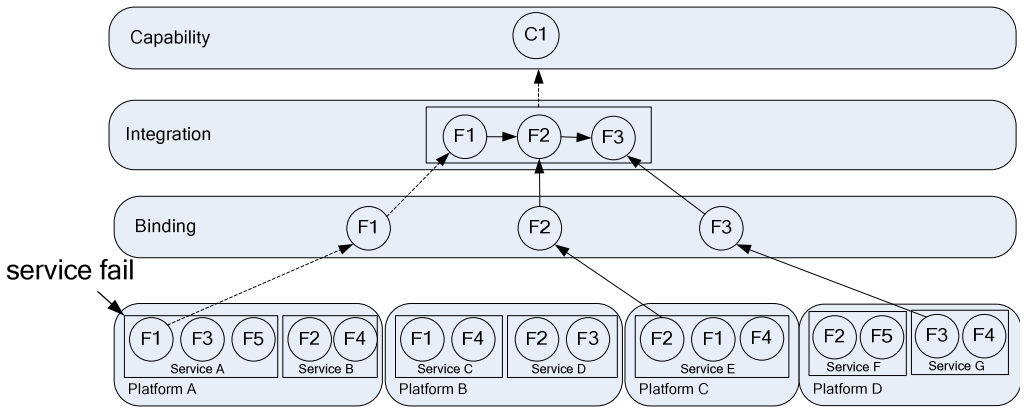


Figure 3: Changes of platform

As shown in Figure 2, services from different platforms can be integrated to deliver a capability. However, when a failure occurs in Service *A* as shown in Figure 3, it could

no longer conform to the requirement in the integration layer to deliver capability *C1*. Capability *C1* would be lost in this case due to the failure of Service *A*. To address this issue, we propose two mechanisms for the reliable and sustainable provision of capability - Redundant Service Binding and Dynamic Service Discovery.

3.2.1 Redundant Service Binding

Redundant service binding is a technique to improve the reliability of the provision of capability. For example, a supplier may need to maintain several aircraft to make one available at any time. The new service development process, which is in the service layer of the architecture shown in Figure 1, is used to develop a new service where further instances of an existing function are needed to achieve the desired level of redundancy.

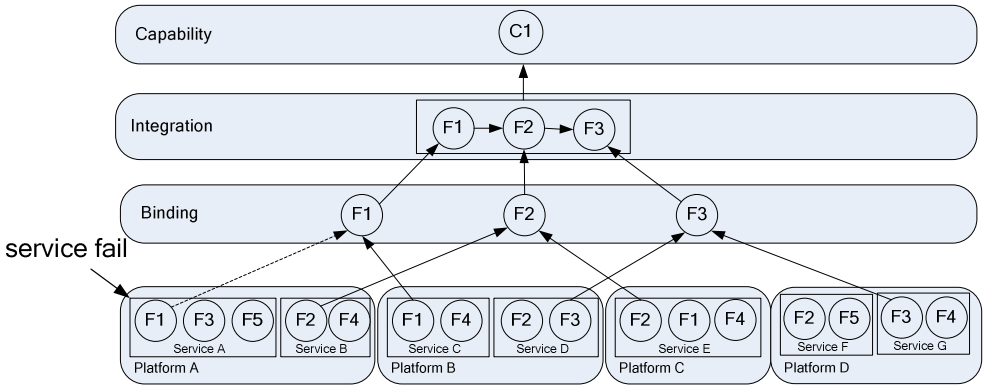


Figure 4: Redundant service binding

In order to provide a reliable capability, the required functions need to be provided by multiple services allocated to different platforms. The reconfiguration algorithm can switch to one of backup services in case of failure of initial service. The distributed recovery block (DRB) scheme (Kim and Welch, 1989) is applied to minimise the reconfiguration time of integration. The DRB scheme is capable of effecting forward recovery while handling both hardware and software faults in a uniform manner. Forward

recovery means that if a failure occurs, the system is restored from an earlier backup. Figure 4 shows an example of redundant service binding. As shown in Figure 4, when a failure occurs in service *A*, the required function provided by the backup service *C* can still work for the provision of capability.

3.2.2 Dynamic Service Discovery

Redundant service binding may increase the reliability of the provision of capability, but the provision of more services may mean higher cost of the provision of a capability and affect affordability. Moreover, redundant services only improve the reliability at a certain time point. They do not handle evolution resulting from ongoing changes. In the example shown in Figure 5, when service *A* fails, the reconfiguration algorithm can switch to the backup service *C* to continue to deliver capability. However, if the backup service *C* fails afterward as shown in Figure 5, the capability is still lost.

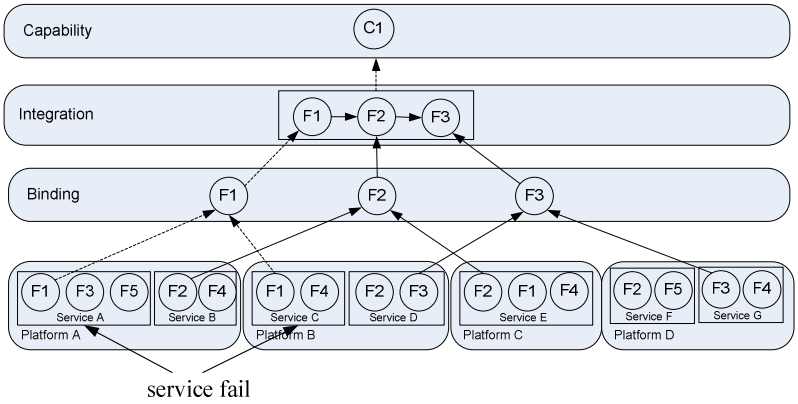


Figure 5: Example of capability loss with redundant service binding

To address this problem, the system should be able to dynamically discover and re-configure new services to provide the requested function to compensate for lost services. Figure 6 illustrates an example of capability reconfiguration with both redundant service binding and dynamic service discovery. When service *A* is not available for use, the

reconfiguration algorithm will not only switch to the backup service *C* to continue to deliver capability, but also simultaneously search the service registry to discover and subscribe to a new service with the requested function *F1* to compensate the lost service. In this case, service *E* is found and bound. When the service *C* fails, service E will automatically take its place and perform the requested function for the provision of capability *C1*.

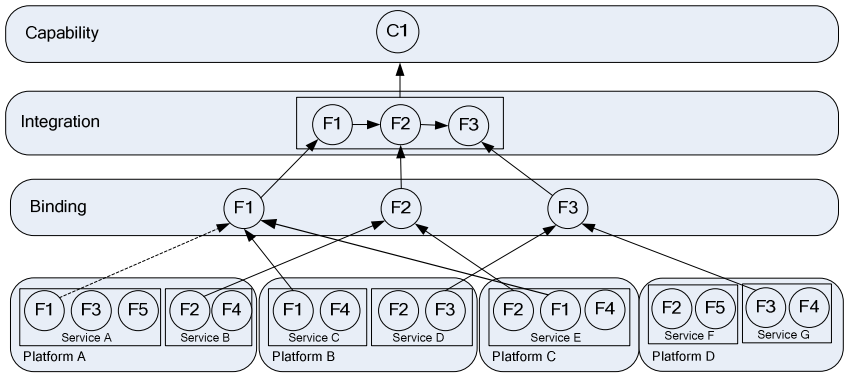


Figure 6: Reconfiguration with redundant service binding and dynamic service discovery

3.3 Evolution of Capability

Apart from evolution of services, evolution of capability can also cause potential problems affecting the reliability of provision of capability. Stakeholders in capability development often change their requirements when the capability has been delivered, in accordance with changes of environment and their needs (Webster et al., 2008a).

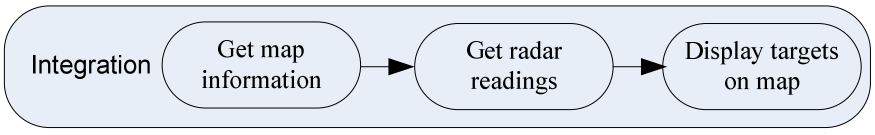


Figure 7: Workflow of service integration

In this section, a region surveillance capability scenario is used as an exemplar to demonstrate the evolution of capability and to analyse the potential influence of capability evolution. We are aiming to get a first insight of how to use an architectural approach to cope with the effect of the evolution of capability for the provision of capability. In a NEC-enabled battlefield, a number of radar sensors supply data through services. The network of radar sensors is modelled conveniently as a dynamic network of services, facilitating ongoing changes. In the modelled system, a surveillance user can submit real-time requests to the system for information of Points of Interest (POIs) in a specified region. A sequence of services (such as "Get map information" and "Get radar reading", "Display targets on map") can be operated in a workflow in order to provide a regional surveillance capability, like the one illustrated in Figure 7.

Existing research work on workflow patterns (Aalst et al., 2003) provides a set of useful tools for analysis and design of workflow for dynamic service integration. A workflow pattern is a form of design patterns specific to the development of workflow applications. Some workflow patterns (Aalst et al., 2003) are listed below:

- Sequence Pattern: An activity in a workflow process is enabled after the completion of another activity in the same process.
- Parallel Split Pattern: A point in the workflow process where a single thread of control splits into multiple threads of control which can be executed in parallel, thus allowing activities to be executed simultaneously or in any order.
- Multiple Choice Pattern: A point in the workflow process where, based on a decision or workflow control data, a number of branches are chosen.
- Simple Merge Pattern: A point in the workflow process where two or more alternative

branches come together without synchronization.

- Synchronizing Merge Pattern: A point in the workflow process where multiple paths converge into one single thread; Synchronization needs to take place if more than one paths are taken.

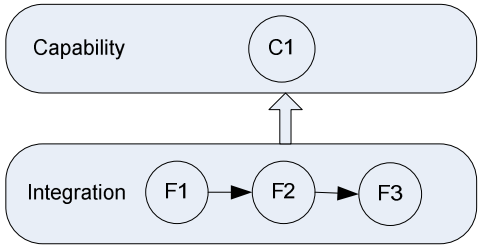


Figure 8: A workflow pattern for a specific delivery of capability

The service integration can be abstracted by using workflow patterns as shown in Figure 8, where *F1* represents the service of getting map information, *F2* represents the service of getting radar reading and *F3* represents the service of displaying targets on map. The capability could be evolved according to changes of environment and users' needs. Two types of possible evolution are illustrated below:

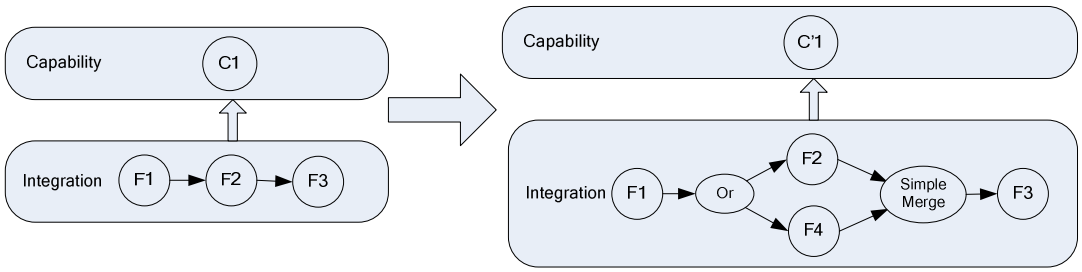


Figure 9: A possible capability evolution (Case 1)

Case 1: In the modelled battlefield, Armed Forces not only have a network of radar sensors which can provide surveillance, but also a number of Unmanned Air Vehicles (UAVs) operating in the area. Information about Points of Interest, POIs, can also be obtained from the UAV sensors. In this case, the original workflow is evolved to a new

version of workflow established on demand with a Multiple Choice pattern (Aalst et al., 2003) as shown in Figure 9, where *F4* represents the service of getting readings from UAVs. The capability could be delivered if either *F2* or *F4*, or both *F2* and *F4*, are successfully executed.

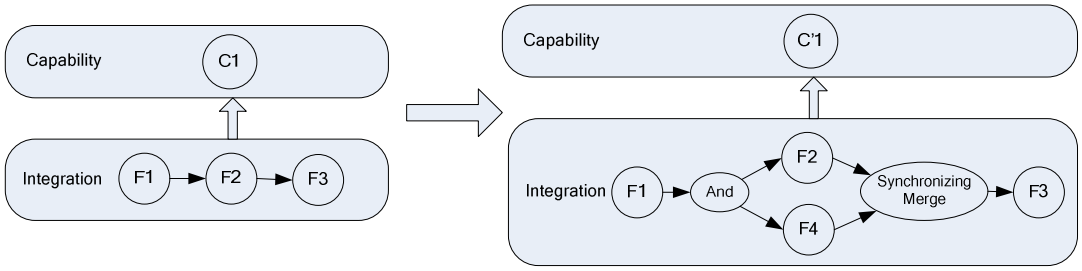


Figure 10: A possible capability evolution (Case 2)

Case 2: It is decided that some POIs are more important and a more aggressive surveillance capability needs to be established. The commander decides to launch a number of UAVs to provide mixed surveillance capability in conjunction with the deployed radars near the battlefield. The capability is evolved with the additional requirement to a new version of capability defined with a Parallel Split pattern (Aalst et al., 2003) illustrated in Figure 10. The new capability could be delivered only in case that both the services: *F2* and *F4* are implemented successfully.

The reliability of provision of service integration could be affected by the evolution of capabilities in both cases mentioned above. In this section, the impact on reliability due to evolution of capabilities is investigated in the architecture layer with a mathematical model. In this model, p is defined as the probability of failing to connect a service for integration. By configuring two services for performing a required function as shown in Figure 4, the probability of failure of a required function for service integration is p^2. In the original case without evolution, three functions are integrated in a workflow

to deliver a capability. Since all three functions are necessary for the provision of a capability, the probability of successful service integration is $\left(1-p^2\right)^3$ in the original example.

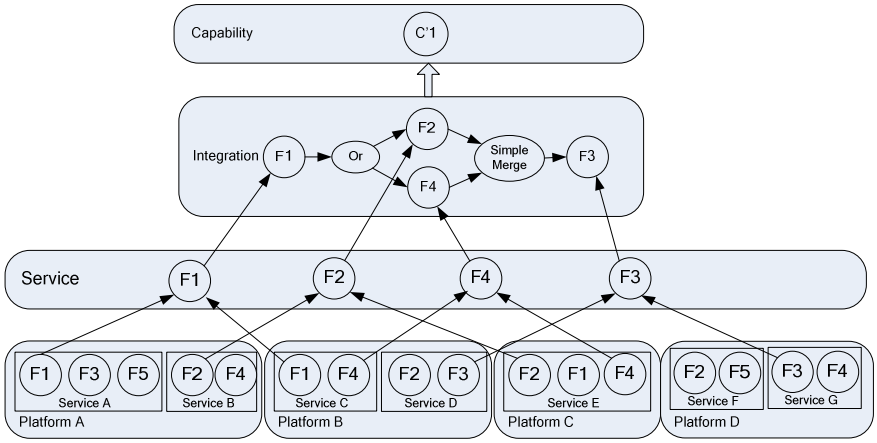


Figure 11: Evolution of capability (Case 1)

In Case 1 as illustrated in Figure 11, the capability can be delivered if either *F2* or *F4* is available. The probability of successful service integration is $\left(1-p^2\right)^2\left(1-p^4\right)$ in Case 1. Since $\left(1-p^2\right)^2\left(1-p^4\right) \geq \left(1-p^2\right)^3 \quad (0 \leq p \leq 1)$, the reliability of provision of capabilities could be improved by the evolution in Case 1.

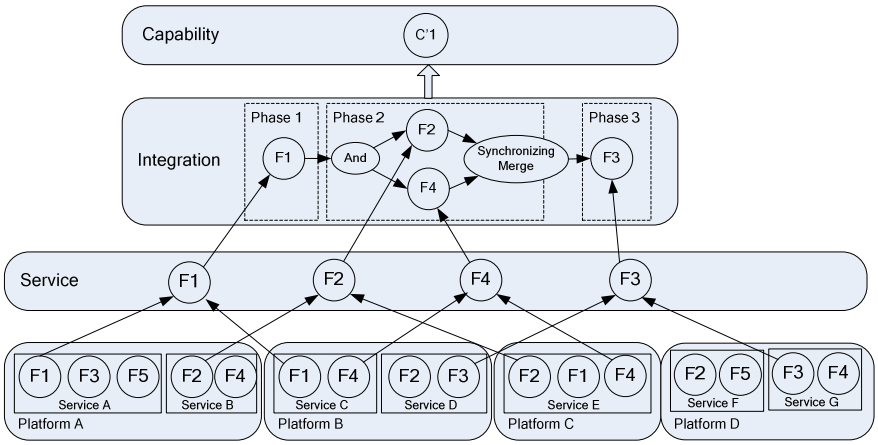


Figure 12: Evolution of capability (Case 2)

In Case 2, the capability could be delivered only when both *F2* and *F4* are available as shown in Figure 12. In Case 2, the reliability of provision of capability is $(1-p^2)^4$. Since $(1-p^2)^4 \le (1-p^2)^3 \quad (0 \le p \le 1)$, the self-diagnosis function identities that the reliability of capability is decreased in Case 2 as shown in Figure 13. For a better understanding of influence of evolution, the workflow in Case 2 is divided into three phases as shown in Figure 12. The success probabilities of the three phases are: $1-p^2$, $(1-p^2)^2$, and $1-p^2$, respectively. The phase 2 is the weak point in the workflow. Its success probability is lower than the original success probability prior to evolution: $P_{evo} = (1-p^2)^2 \le P_{org} = 1-p^2 \quad (0 \le p \le 1)$. To address this issue, the self-diagnosis function invokes self-configuration function to proactively modify its behaviour to self-adapt the evolution of capability as shown in Figure 13.

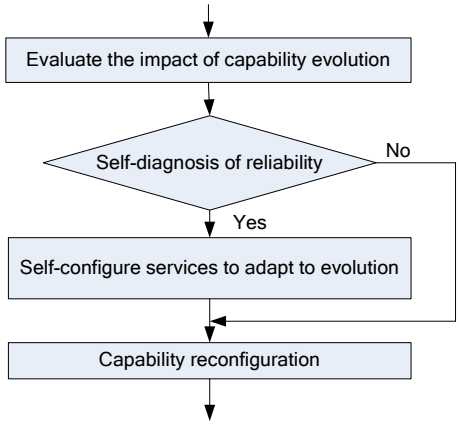


Figure 13: self-adaptive configuration for provision of capability

The redundancy needs to be dynamically self-justified, in order to cope with the impact of service integration. As discussed above, the success probability of the second phase is evaluated as $P_{evo}^{R} = (1-p^R)^2$, where R is redundancy of service binding which is

defined by the number of services bound for performing a required function to deliver a capability. For sustainable provision of capability, more services ($R > 2$) need to be added and configured to provide each function *F2* and *F4* to enable its Cumulative Distribution Function (CDF) smaller than the original CDF prior to the evolution, where *x* defined as the probability of failing to connect a service for integration

$$\int_0^1 \left(1 - x^R\right)^2 dx \geq \int_0^1 (1 - x^2) dx . \qquad (1)$$

The inequality (1) is satisfied only in the case of $R \geq 4$. More services configured mean higher cost. In this case, the minimum value $R = 4$ is adopted to minimise the cost of the sustainable provision of capability as illustrated in Figure 14.

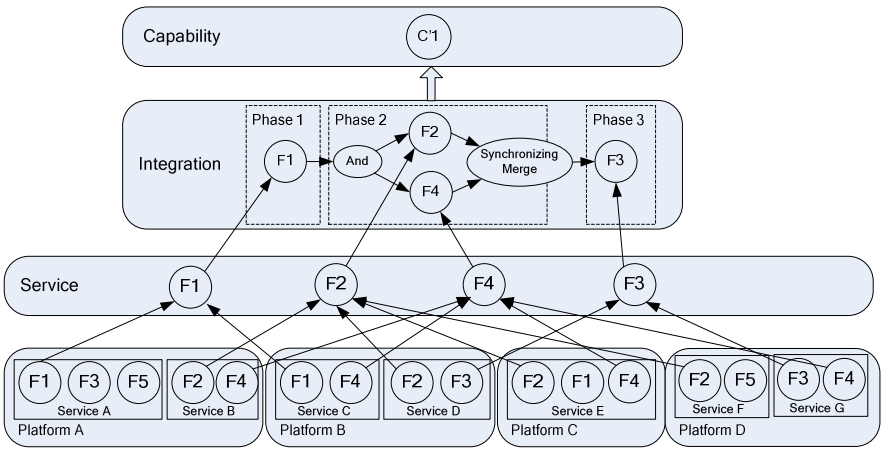


Figure 14: Evolution (Case 2) with adaptive service reconfiguration

This probabilistic model discussed above can be applied to different cases. The aim of the model is to reconfigure services to maintain the reliability of capability provision, where the reliability after the evolution of capability should be no less than the reliability of original configuration. Minimum redundancy *R* is suggested to minimise the influence of the affordability of the capability provision.

4. Evaluation

In this section, the reliability and performance of provision of capability on SOA are evaluated using simulations to see whether the architectural approaches described above, can achieve enhanced performance for the delivery of network enabled capability. Dependable dynamic service integration enables services to be integrated both dependably coping with evolution of services and requirements and dynamically at runtime across departments and organisations. The two architectural approaches for dependable dynamic service integration for the delivery of networked enabled capability were simulated in different situations: (1) an SOA-based approach of the mechanisms of redundant service binding and dynamic service discovery to cope with evolution of services in the provision of capability; and (2) a self-adaptive approach that is able to dynamically evaluate the influence of evolution of requirements and self-configure services to adapt to the evolution of capability.

4.1 Simulation Setup

The simulation model has been developed using the Java programming language. The main components of the simulation model are illustrated in Figure 15.

Different systems have different topologies. But many of them are evolving from random topology (Watts, 2005). The simulation starts from random platform composition and topology to see the influence of capability evolution. In the simulations, a network was setup containing thirty platforms (e.g., ten major sea platforms and twenty air platforms). Fifteen different high level functions were generated and each service performed three functions. Each platform provided five services which were randomly

selected from a pool of 100 services. Each platform randomly connected to four other platforms bi-directionally and formed a random topology.

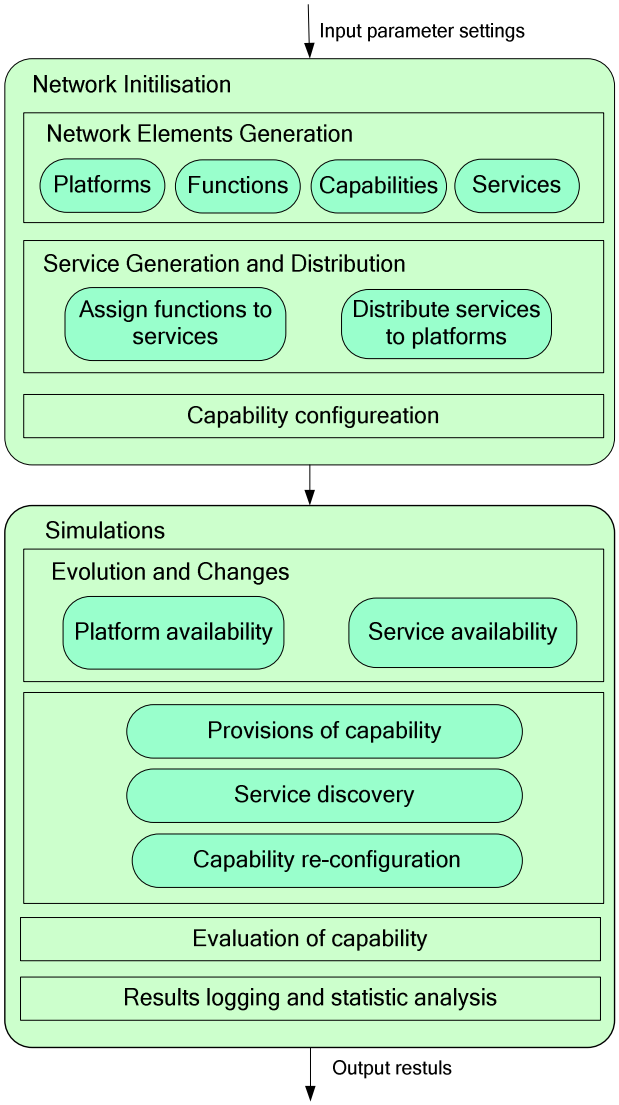


Figure 15: Simulation model

As noted above, ongoing changes could be caused by adding and removing services from platforms. To simulate the evolution of platforms, one platform was randomly selected and upgraded to provide one extra service to the network and update one platform to remove one previously provided service from the network every hour in a

simulation loop. The availability of each platform was set at 70% in all simulations. In the simulations, two services ($R = 2$) providing a required function were bound and configured as illustrated in Figure 4, if no other setting is mentioned.

In response to this need, simulations employing timing parameters were carried out to show the performance of delivering real-time capability on SOA. Based on (Martinello et al., 2003, Powell, 2003), the service response time, the delay of error detection, rollback and switching a backup service were set at 3.2 seconds, 2 seconds, 1 second and 2 seconds, respectively. In contrast to service binding, the process of new service discovery, verification (model checking) and validation (testing) (Tsai et al., 2006b) is much more complex and time consuming. A longer delay of the process (100 seconds) was set up in the simulations.

4.2 Simulation Results

Owing to many customised functions offered, there are many combinations of parameters to experiment with, which could generate far too many graphs to analyse. In this section, we only present an analysis of simulation results from the most pertinent experiments as we see.

4.2.1 Effect of Evolution of Services

4.2.1.1 Redundant Service Binding

As noted above, redundant service binding could be one way to improve the reliability of the provision of capability. In this section, the reliability of capability provision is compared by means of using different redundancy *R*. The simulation parameters changed for this experiment are shown in Table 1.

Table 1. Changes to parameters for simulation

Parameter	Value
R	1; 2; 3; 4

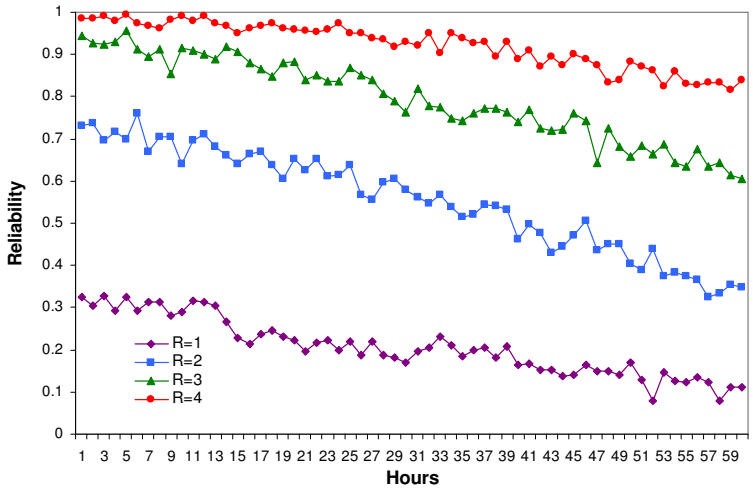


Figure 16: Reliability of capability provision with redundant service binding

Figure 16 shows the reliability of capability provision in the networks where one, two, three and four services providing a required function are bound and configured for the provision of a capability, respectively. As shown in Figure 16, redundant services increase the reliability of the provision of capability. The capability configured with the highest redundancy ($R = 4$) achieves the highest reliability. The reliability of capability provision is not constant as a function of time but decreases due to platform failure causing loss of services and functions.

4.2.1.2 Dynamic Service Discovery

In this experiment, we examine the reliability of capability provision with redundant service binding ($R = 2$) and dynamic service discovery (as described in Section 3.2.2),

and compare its reliability with capability provision with redundant service binding only ($R = 2$), as described in Section 3.2.1.

Figure 17 shows the reliability of capability provision using dynamic service discovery with redundant service binding. As shown in Figure 17, sustainable provision of capability with high reliability is achieved with redundant service binding and dynamic service discovery in the simulation environment ($R = 3$), since new services have been dynamically discovered to compensate the loss of services. Dynamic changes caused by evolution of network and platforms have been mostly handled in this case.

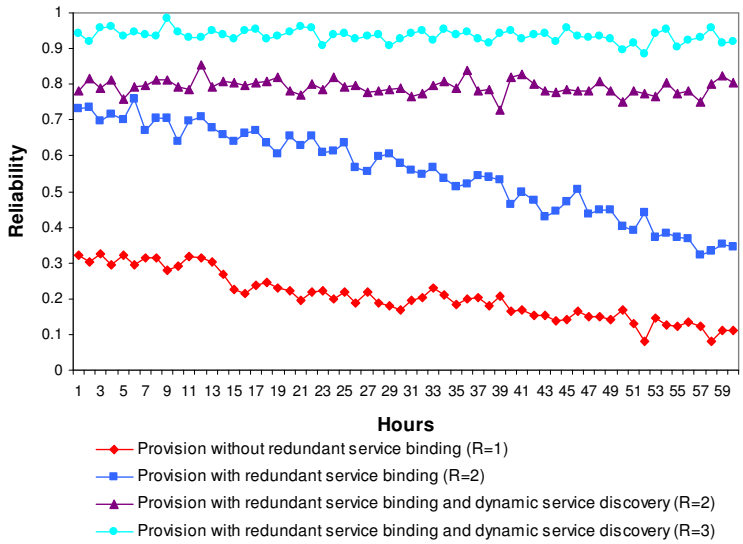


Figure 17: Reliability of capability provision with redundant service binding and dynamic service discovery

From the results shown in Figure 17, the reliability increases by 89% by changing R from 1 to 2. However, reliability grows by only about 19% by modifying R from 2 to 3. Since dynamic service discovery with the highest redundancy achieves the highest reliability, multiple services ($R \geq 3$) providing each required function need to be configured to deliver a critical capability with high assurance requirements. But the

provision of more services could lead to higher cost and affect affordability. In contrast, the redundancy $R = 2$ could be considered for the development of non-critical capability, which can achieve a significantly improved reliability (compared to $R = 1$) with comparable cost.

4.2.1.3 Time-constrained capability provision

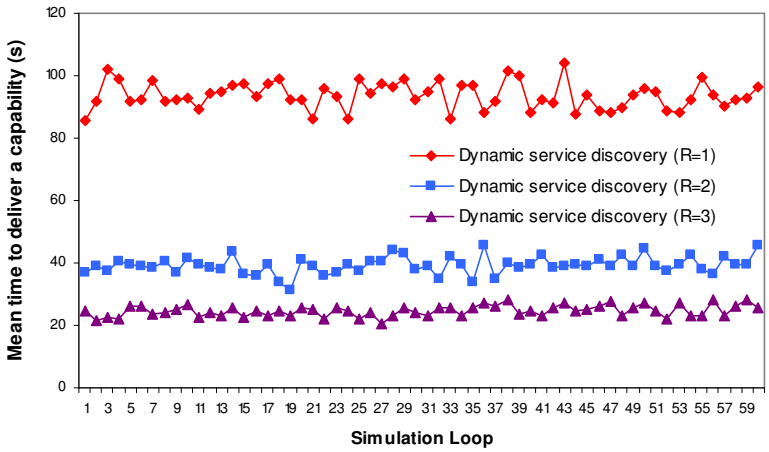
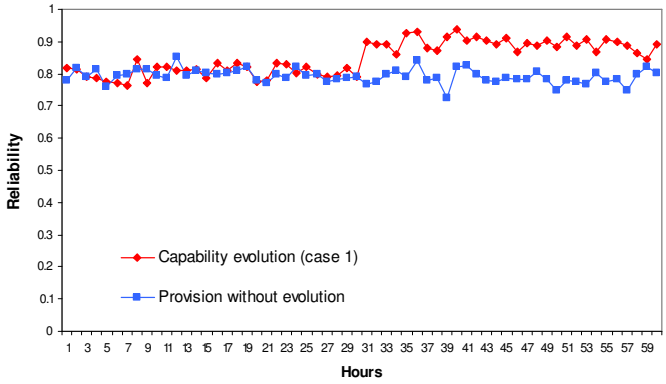


Figure 18: Mean time to deliver a capability

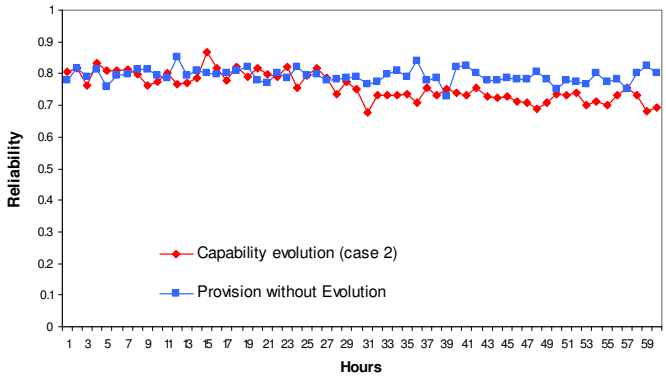
Figure 18 shows the simulation results of mean time to deliver a capability. As shown in Figure 18, redundant service binding significantly reduces the capability delivery time which is reduced by 63% by changing R from 1 to 2. But additional redundant service ($R = 3$) contribute little to the further reduction of time. The result suggests that redundant service binding is essential for delivering a real-time capability in a dynamic environment and the sustainable real-time capability can be achieved with our architectural approach.

4.2.2 Effect of Evolution of Capability

In this section, simulations were carried out to show the effect of evolution of service integration. Two types of evolution: Case 1 and Case 2 (as discussed in Section 3.3) have been injected into the simulations at the 30th hour.



(a)



(b)

Figure 19: Reliability with evolution in (a) Case 1; (b) Case 2

Figure 19(a) shows the reliability of provision of capability with the evolution in Case 1. From the results shown in Figure 19(a), we can see that reliability increases by the evolution of capability with the positive impact of the evolution as we analysed in Section 3.3. Figure 19(b) shows the reliability of provision of capability with the evolution in Case 2. As shown in Figure 19(b), the evolution of capability in Case 2 negatively affects the reliability.

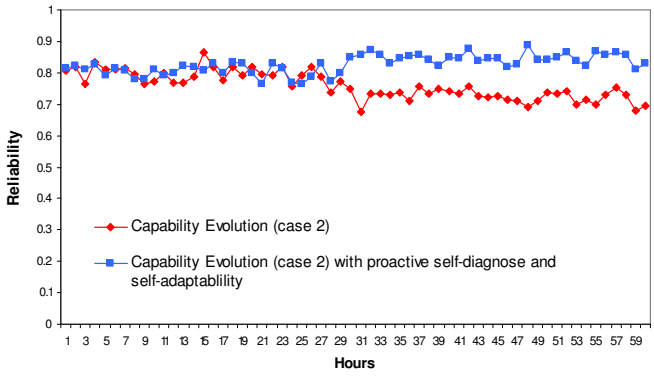


Figure 20: Provision of capability with adaptive re-configuration (Case 2)

Figure 20 shows the reliability of provision of capability with proactive reconfiguration introduced in Section 3.3.1. Once the evolution of capability occurred at the 30th hour, the reconfiguration algorithm of the new model can autonomously identify the impact of the evolution and self-adapt the redundancy of *F2* and *F4* to $R = 4$ accordingly. From the results shown in Figure 20, the negative impact of evolution in Case 2 has been handled by the proactive self-diagnostic and self-adaptive technique. The sustainable provision of capability has been achieved through dynamic service integration and reconfiguration.

5. Case Study: Region Surveillance

The reliability and performance of provision of capability on SOA have been verified through modelling and simulations. A NEC demonstration system has been developed and tested to further ascertain that the implementation of the SOA approach is suitable for use in a real case. This system shows the dynamic service integration of a network of sensors on a battlefield to provide a reliable regional surveillance capability. The core of the approach is the process of mapping high-level requirements for capability onto the

invocation of actual services. This approach allows the establishment of a dynamic workflow of service composition (Figure 21) and dynamic search for services and on the fly planning through dynamic integration of services. The competitive advantage, such as timeliness, reliability and fault tolerance, can be achieved through the dynamic service discovery, composition and integration.

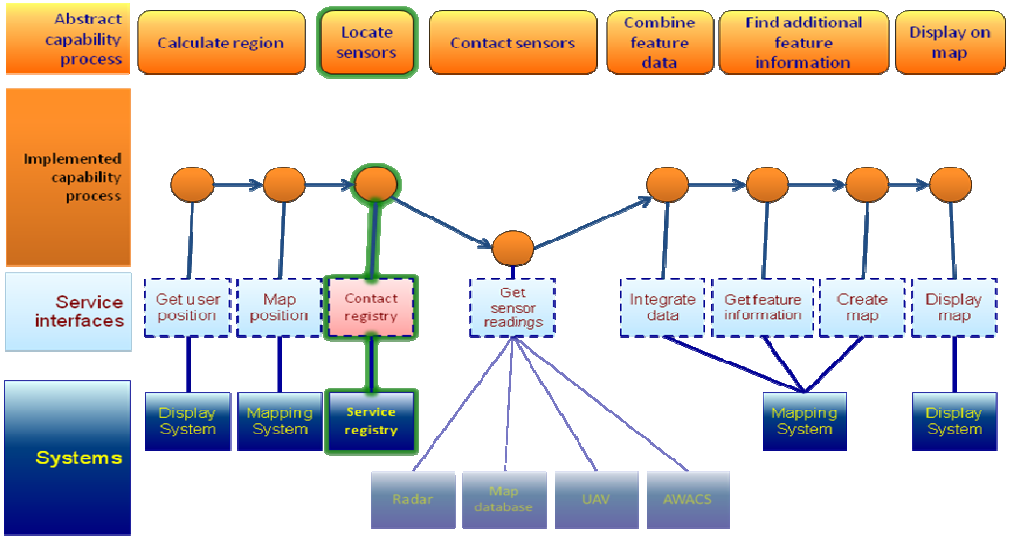


Figure 21: Dynamic workflow of service composition

The intent is to demonstrate the architectural approach to engineering and using systems in NEC. The main concepts are:

- Use of SOA in NEC enhanced with other architectural styles and patterns;
- Integration of distributed systems in a dynamic environment;
- Coping with changes in availability of distributed components;
- Evolution of the systems that provide service implementations;

In the NEC-enabled battlefield as used in the demonstrator, sensors can supply data through services, and such a network of sensors can be modelled conveniently as a dynamic network of services, facilitating ongoing changes.

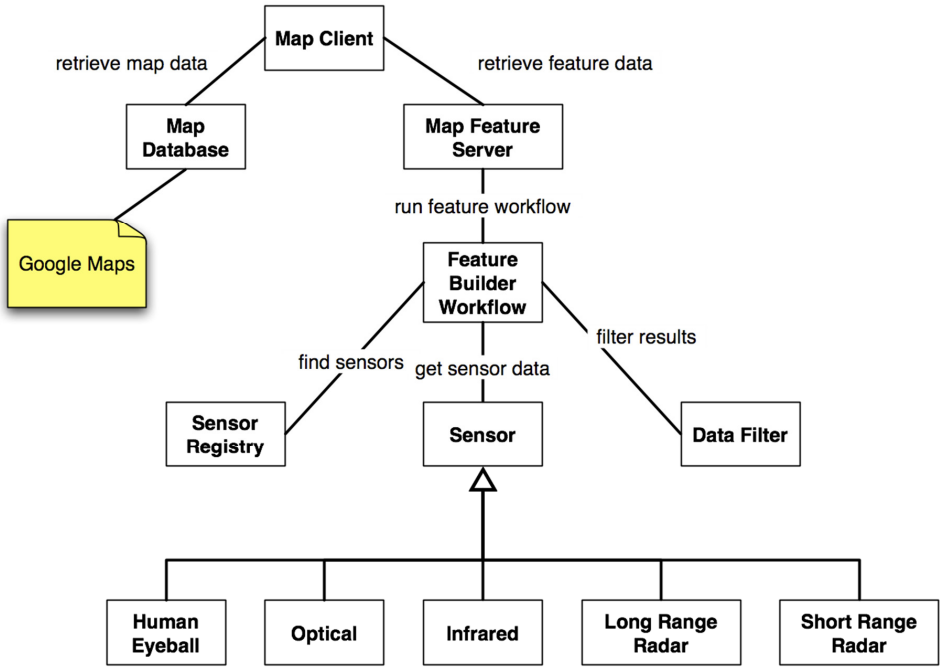


Figure 22: System Architecture

The architecture of the system is shown in Figure 22. In the system, a surveillance user can submit real-time requests to the system for information of Points of Interest, POIs, in a specified region. POIs include but are not limited to troops, land vehicles, communication and weapon systems, as well as buildings, bridges, and other static objects in the environment. Surveillance data is provided through a network of sensors of different types, such as human eyeball, visible, infrared optical, long and short-range radar. The system dynamically discovers sensors, retrieving attributes such as position and range. A selection algorithm determines which sensors can 'see' the region of interest (ROI). The relevant sensors are contacted, which return information about the detected

POI. The system returns the related information about the POIs within that region, e.g., current locations of those POIs (Figure 23).

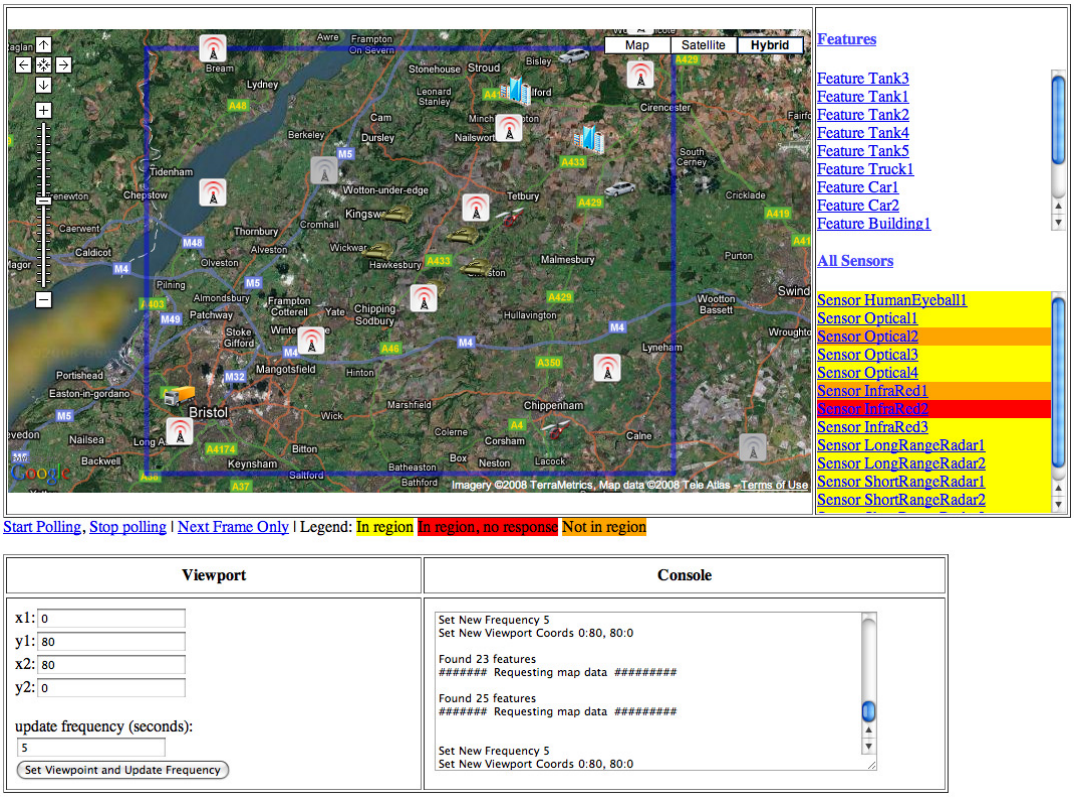


Figure 23: Region surveillance showing Points of Interest

The system is built on a dynamic and changing environment, where sensor services in region may fail to respond with information about the POIs as shown in Figure 23. By using the approach proposed in Section 3, multiple sensor services are contacted to receive the data about POIs in the requested region as shown in Figure 21. The system is used to illustrate aspects of the research into systems architecture and through-life systems management (TLSM).

Simple data can be collected from individual services, while complex data is generated through composition of multiple services. The possibility and quality of on-the-fly planning and application construction largely depend upon

- correct interpretation of user requirements,
- information available on services,
- matching between requirements and services, and
- interoperability between services.

In contrast to a standard SOA approach, the demonstration system incorporates the following innovations to achieve competitive advantage:

- *Information-Rich Information Services*: provide description of services, composition templates with candidate composed services, application workflows, architectural patterns, application patterns, evaluation information (Tsai et al., 2008).
- *Evolving Ontology*: ontology available for dependability, capability, system assessment (Webster et al., 2008b).
- *Service Interoperability*: advanced techniques for dynamic authentication and run-time negotiation (Townend et al., 2008).
- *Optimisation for On-the-Fly Planning*: based on a tool (Townend et al., 2008) that supports the use of a variety of optimization techniques and their combination.

6. Conclusion and Future Work

In this paper, we have presented two innovative architectural approaches using the concepts of evolutionary service-oriented architecture (Liu et al., 2008b) and dynamic workflow management to enable dynamic service integration for provision of reliable and

sustainable military capability, including a SOA-based approach of the mechanisms of redundant service binding and dynamic service discovery to cope with evolution of services in the provision of capability, and a self-adaptive approach that is able to dynamically evaluate the influence of evolution of capability and self-configure services to adapt to the evolution of capability. These approaches are high-level architectural approaches for constructing complex software-intensive systems, which contributed to Decision Support for NEC (Whitfield et al., 2007) for delivering essential information across multiple networked resources and Control and Monitoring for NEC (Yao et al., 2007) for the development of appropriate tools for autonomous management of manned and unmanned assets.

These approaches have been verified through modelling and simulation of SOA for NEC and demonstrated through developing and testing the NEC demonstration system for a region surveillance capability scenario. The simulation results show that our architectural approaches provide a high-level of reliability and sustainability in handling dynamic changes and evolution that would be encountered in the delivery of military capability. The development of the NEC demonstration system has been used to ascertain that the implementation of the SOA-based architectural approach is fit for use.

In our future work, the probabilistic model used in this paper will be further extended to cover also more general cases. The implementation in the real-world systems will be carried out in the next step for generalised quantitative validation. Further development of the demonstrator will be used for further evaluation of evolutionary SOA and NEC systems. The investigation will link to lifecycles for service delivery and agile

methods to respond to changes owing to, for example, faults, customer need, technology developments and obsolescence.

7. Acknowledgement

The work reported in this paper has been supported by the U.K. EPSRC Platform Grant: WRG Phase III EP/F0576644/1 and the NECTISE programme jointly funded by BAE Systems and the U.K. EPSRC Grant EP/D505461/1.

References

Russell, D. & Xu, J. (2007) Service Oriented Architectures in the Provision of Military Capability. *UK e-Science All Hands Meeting*.

Alonso, G., Casati, F., Kuno, H. & Machiraju, V. (2004). *Web Services Concepts, Architectures and Applications*: Springer Verlag.

Tsai, W.-T., Zhou, X. & Chen, Y. (2006a). PESOI: Process Embedded Service-Oriented Architecture. *Journal of Software* 17: 1470–1484.

DoD (1994). *DoD Dictionary of Military and Associated Terms*: United States Department of Defense Joint Publication.

Liu, L., Russell, D., Xu, J., Davies, J. & Irvin, K. (2008a) Agile Properties of Service Oriented Architectures for Network Enabled Capability. *Realising Network Enabled Capability (RNEC'08)* Leeds, United Kingdom.

UK Ministry of Defence (2005a) Defence Industrial Strategy: Defence White Paper (CM6697).

UK Ministry of Defence (2005b) Joint Services Publication 777, Edition 1, p. 1.

Liu, L., Antonopoulos, N., Mackin, S., Xu, J. & Russell, D. (2009). Efficient Resource Discovery in Self-organized Unstructured Peer-to-Peer Networks. *Concurrency and Computation: Practice and Experience* 21: 159–183.

Avizienis, A., Laprie, J.-C., Randell, B. & Landwehr, C. (2004). Basic Concepts and Taxonomy of Dependable and Secure Computing. *IEEE Transactions on Dependable and Secure Computing* 1.

Liu, L., Russell, D., Looker, N., Webster, D., Xu, J., Davies, J. & Irvin, K. (2008b) Evolutionary Service-based Architecture for Network Enabled Capability. *International Workshop on Verification and Evaluation of Computer and Communication Systems* Leeds, United Kingdom.

OASIS (2006) Web Services Composite Application Framework (WS-CAF).

Casati, F., Ilnicki, S., LiJie Jin, Krishnamoorthy, V. & Shan, M.-C. (2000) Adaptive and dynamic service composition in eFlow. *12th International Conference on Advanced Information Systems Engineering(CAiSE)*.

Casati, F. & Sha, M.-C. (2001) Dynamic and adaptive composition of e-services. *12th International Conference on Advanced Information Systems Engineering (CAiSE 00)*.

Andrews, T., Curbera, F., Dholakia, H., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I. & Weerawarana, S. (2003) Specification: Business Process Execution Language for Web Services Version 1.1. IBM, BEA Systems, Microsoft, SAP AG, Siebel Systems.

Tsai, W.-T., Zhou, X., Chen, Y. & Bai, X. (2008). On Testing and Evaluating Service-Oriented Software. *IEEE Computer* 41: 40-46.

Russell, D., Looker, N., Liu, L. & Xu, J. (May 2008) Service-Oriented Integration of Systems for Military Capability. *IEEE International Symposium on Object/component/service-oriented Real-time distributed Computing* Orlando, Florida.

Webster, D., Russell, D., Looker, N. & Xu, J. (2008a) Motivations for Change within an Enduring Architecture. *Realising Network Enabled Capability (RNEC'08)* Leeds, UK.

Kim, K. H. & Welch, H. (1989). Distributed Execution of Recovery Blocks: An Approach for Uniform Treatment of Hardware and Software Faults in Real-Time Applications. *IEEE Transactions on Computers* 38: 626-636.

Aalst, W. v. d., Hofstede, A. t., Kiepuszewski, B. & Barros, A. (2003). Workflow Pattern. *Distributed and Parallel Databases* 14: 5-51.

Watts, D. (2005). *The Dynamic of Networks Between Order and Randomness*: Princeton University Press.

Martinello, M., Kaˆaniche, M. & Kanoun, K. (2003) Web Service Availability – Impact of Error Recovery. *The International Conference on Computer Safety, Reliability and Security* Edinburgh, UK.

Powell, M. (2003) At Your Service: Performance Considerations for Making Web Service Calls from ASPX Pages. In: MSDN Article, Microsoft Corporation.

Tsai, W.-T., Lee, Y.-H., Cao, Z., Chen, Y. & Xiao, B. (2006b) RTSOA: Real-time Service-Oriented Architecture. *the Second IEEE International Symposium on Service-Oriented System Engineering (SOSE'06)* pp. 49-51). Shanghai.

Webster, D., Looker, N., Russell, D., Liu, L. & Xu, J. (2008b) An Ontology for Evaluation of Network Enabled Capability. *Realising Network Enabled Capability (RNEC'08)* Leeds, United Kingdom.

Townend, P., Huai, J., Xu, J., Looker, N., Zhang, D., Li, J. & Zhong, L. (2008). CROWN-C: A High-Assurance Service-Oriented Grid Middleware System. *IEEE Computer* 41: 33-38.

Whitfield, R. I., Duffy, A., Thomson, A., Wu, Z. & Liu, S. (2007) An Architecture for Organisational Decision Support. *Systems Engineering for Future Capability conference*.

Yao, L., Gu, D. & Postlethwaite, I. (2007) A Simulation Environment for Control and Monitoring Schemes in Network Enabled Systems. *Systems Engineering for Future Capability conference*.

Dr Lu Liu is Lecturer in School of Engineering and Information Sciences at Middlesex University (UK). Before joining Middlesex University, he was Research Fellow in the School of Computing at the University of Leeds (UK), working on NECTISE Project which was an UK EPSRC/BAE Systems funded research project involving ten UK Universities and CoLab Project which was funded by UK EPSRC and China 863 Program. He received a Ph.D degree (funded by UK DIF DTC) from the University of Surrey (UK) and M.Sc. degree from Brunel University (UK). His research interests are in areas of peer-to-peer computing, software engineering and service-oriented computing. Dr Liu has over 30 scientific publications in reputable journals, academic books and international conferences. He won the Best Paper Award at the Realising Network Enabled Capability Conference in 2008. He is on the Editorial Review Board of International Journal of Distributed Systems and Technologies. He served as Co-chair, TPC Member, Advisory Committee Member and Session Chair of many international conferences and workshops. He is a member of IEEE.

Professor Jie Xu is Chair of Computing at the University of Leeds (UK), Director of the Institute for Computational and Systems Science at Leeds, and Director of the EPSRC WRG e-Science Centre involving the three White Rose Universities of Leeds, York and Sheffield. He has worked in the field of Distributed Computer Systems for over twenty-five years and had industrial experience in building large-scale networked systems. Professor Xu now leads a collaborative research team at Leeds studying Internet and

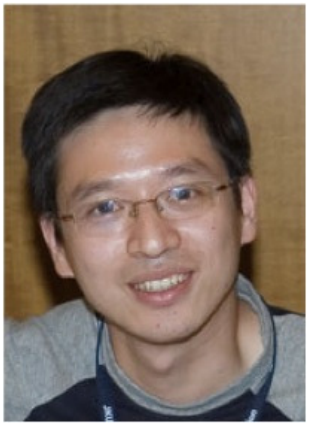
Cloud technologies with a focus on complex system engineering, system security and dependability, and evolving system architectures. He is the recipient of the BCS/IEE Brendan Murphy Prize 2001 for the best work in the area of distributed systems and networks. He has led or co-led many key research projects, and published more than 280 academic papers and edited books. Professor Xu has served as general Chair/Program Chair/PC member of numerous international computer conferences and has been an editorial board member for several international Computing journals.

Dr Duncan Russell is a Senior Solutions Architect at Image Analysis Ltd. Dr Russell was previously a Senior Research Fellow in the Distributed Systems and Services Group, School of Computing at the University of Leeds, leading the Systems Architecture topic in NECTISE which is a project jointly funded by BAE Systems and EPSRC. He completed his PhD thesis on Secure Service-based Collaborative Workflow Across Organisations as part of the DAME project under Professor Peter Dew at Leeds in 2007. Dr Russell gained a BEng in Electronic Systems Engineering at Kingston University in 1994. Following this he worked on cordless communications at Philips Research Laboratories, UK and he spent 6 years working in R+D at BSS Audio, designing professional audio equipment (software and hardware) for install and live audio markets. During his time at BSS Audio, he led the project for Soundweb embedded products.

Dr John K Davies is an Engineering Manager at BAE Systems Integrated Systems Technology and a visiting professor at the University of Leeds. John received his BSc in physics from the University of Bristol and PhD from University College London. He then carried out physics research for the Rutherford Laboratory and Oxford University on

experiments at CERN, Geneva and Fermilab, Chicago, before switching to a career in engineering. John has worked as a software and systems engineer and manager on a number of defence projects and studies and is currently responsible for the processes and tools used to carry out and support systems engineering. He has over sixty referenced publications in physics and engineering. John is a Chartered Engineer, Fellow of the Institute of Engineering and Technology, and a member of INCOSE.

Dr David Webster is currently a PhD candidate student working on his second PhD within the Distributed Systems and Services group. He is working in the area of providing dependable interface mediator technology for evolving Web Service based 'System of Systems'. From 2006-2009 David was previously a research assistant for the University of Leeds Distributed Systems and Services group, primarily working on the now completed NECTISE project with BAE Systems. Within NECTISE, David worked on Life-cycle modelling for 'System of Systems' engineering to support Network Enabled Capability (NEC). David has successfully defended his first PhD's viva at the University of Hull, subject to minor amendments. The thesis submitted is for a context-oriented concept expansion and comparison framework using Wikipedia as a common sense knowledge-base. David holds professional membership (MBCS) status within the BCS and is a member of the International Council on Systems Engineering (INCOSE).



Dr Zongyang Luo is currently with Jinton Group, P.R.China. He was a post doctor researcher at school of computing in the University of Leeds. His research interests

include software and system architecture, electronic commerce, distributed systems, wireless networks, performance engineering. He is a member of IEEE. He received his PhD degree in Electronic Engineering from the University of Surrey, UK.

Dr Colin C. Venters is an assistant professor at Kyungpook National University, South Korea. His research interests include requirements engineering with a particular emphasis on architectural-level reasoning, multimedia information retrieval, distributed systems, information visualization and the Semantic web. He is a member of the ACM and IEEE. Dr Venters received his Ph.D in computer science from the University of Manchester.