



Magazine

Systemy wielosystemowe

Autor: Rafał Gajdulewicz

O autorze: Absolwent Wydziału Matematyki i Nauk Informatycznych Politechniki Warszawskiej. Obecnie pracuje w firmie SMT Software na stanowisku Software Developera i kończy studia na Wydziale Nauk Ekonomicznych Uniwersytetu Warszawskiego.

Kontakt: r.gajdulewicz@gmail.com



Advanced

Level

5

Magazine Number

Testowanie oprogramowania

Section in the magazine

Wprowadzenie

System wielosystemowy (ang. system-of-systems) jest stosunkowo nowym pojęciem w inżynierii oprogramowania. Oznacza ono system zbudowany na zasadzie współpracy dużych systemów, zarówno istniejących wcześniej, jak i wytwarzanych specjalnie na potrzeby kooperacji. Wytwarzanie tak złożonych systemów obarczone jest dużym ryzykiem błędu, a ocena jakości powstającego produktu jest kluczowa dla powodzenia projektu.

Aby dostarczyć wymaganych informacji o jakości, niezbędne jest właściwe zaplanowanie i przeprowadzenie procesu testowania zarówno systemów komponentowych, jak i całości systemu wielosystemowego. Testowanie tak złożonych systemów wymaga zastosowania innego podejścia niż w przypadku klasycznych, pojedynczych systemów.

Artykuł ten jest przeglądem zagadnienia testowania systemów wielosystemowych, opisanego bardziej szczegółowo w pracy magisterskiej autora, dostępnej pod adresem http://coremag.eu/fileadmin/Papers/gajdulewicz_Metody_testowania_systemow_wielosystemowyc_h.pdf. Jej celem był przegląd literatury dotyczącej systemów wielosystemowych, opis problemów związanych z ich testowaniem, a także sposobów radzenia sobie z nimi.

Praca opierała się na udostępnionych publicznie opisach przypadków i metod testowania systemów wielosystemowych, wytwarzanych głównie na zlecenie Departamentu Obrony USA. Podstawowymi źródłami informacji na temat testowania systemów wielosystemowych były prace Bernarda Homes'a oraz Rolanda Brooks'a i Andrew Sage'a.

Systemy wielosystemowe - definicja

Poniższa definicja pochodzi z Certified Tester Advanced Level Syllabus i definiuje system wielosystemowy jako:

Zbiór współpracujących komponentów (sprzęt, poszczególne aplikacje i metody komunikacji) połączonych w celu osiągnięcia wspólnego rezultatu, nie posiadających jednolitej struktury zarządzania

Szczególnie istotnym zarówno dla wytwarzania, jak i dla testowania stwierdzeniem jest „[...] nie posiadających jednolitej struktury zarządzania”. Jest to jedna z najbardziej oczywistych cech odróżniających systemy wielosystemowe od klasycznych systemów.

Alternatywna definicja, pochodząca z http://en.wikipedia.org/wiki/System_of_systems podkreśla z kolei inną istotną właściwość systemów wielosystemowych - fakt, że dostarczana przez nie funkcjonalność jest znacząco większa od sumy funkcjonalności poszczególnych komponentów:

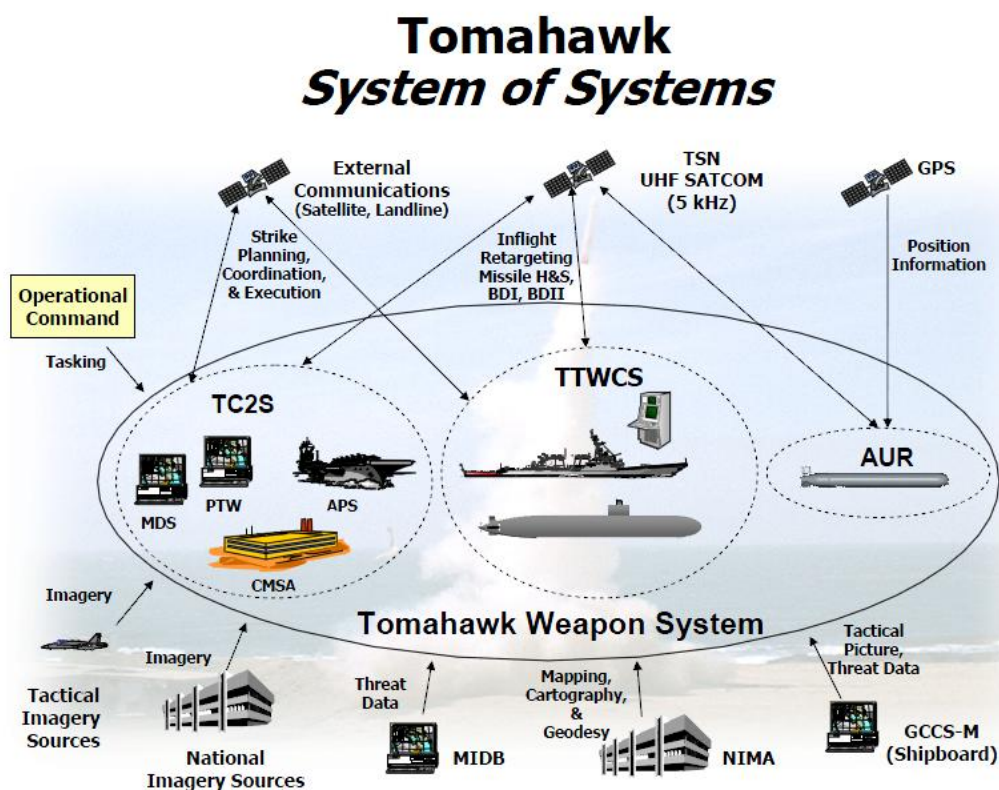
Zbiór zorientowanych zadaniowo lub dedykowanych systemów, które współdzielą zasoby i możliwości by stworzyć nowy, bardziej złożony "meta system", który przewyższa funkcjonalnością i wydajnością sumę użytych systemów

Przykłady

Systemy wielosystemowe wraz z rozwojem technologicznym znajdują zastosowanie w coraz większej liczbie dziedzin, z których kilka najpopularniejszych zostało omówionych poniżej. Wśród nieomówionych klas systemów wielosystemowych można jeszcze wymienić systemy medyczne, systemy nadzoru przestrzeni powietrznej czy zintegrowane systemy transportowe.

Systemy kontroli pojazdów

Pierwszą omawianą klasą systemów wielosystemowych są systemy kontroli pojazdów i pocisków powietrznych, wśród których klasycznym przykładem jest system obsługi pocisków Tomahawk. Został on wprowadzony do użycia w roku 1983, a po raz pierwszy został zastosowany w walce w roku 1991 podczas konfliktu w Zatoce Perskiej. Od tamtej pory przeszedł kilka istotnych modyfikacji, dotyczących nie tylko samego pocisku, ale także sposobu kontroli nad nim i źródeł danych wykorzystywanych do namierzania celu.



Rysunek 1 Schemat systemu Tomahawk

Źródło: Jeffrey S. Mayer "Tactical Tomahawk Weapon System"

System Tactical Tomahawk jest ciekawy nie tylko jako wzorcowy wręcz przykład systemu posiadającego wszelkie cechy systemu wielosystemowego, ale także z powodu zmian w kluczowych elementach jego struktury, takich jak system sterowania i pobierania danych o położeniu.

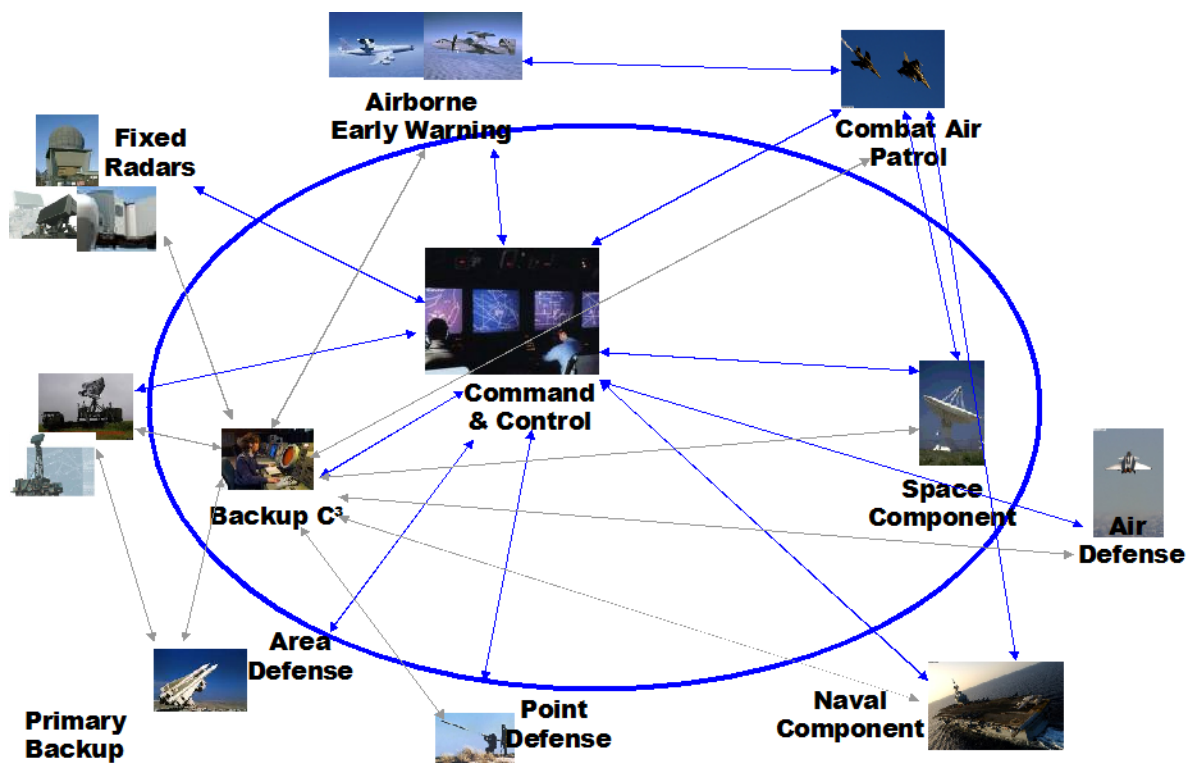
System sterowania pociskiem ewoluował od systemu TWCS, będącego adaptacją starego systemu sterowania czołgami, poprzez system ATWCS, należący do klasy komercyjnego oprogramowania z

półki (ang. commercial-off-the-shelf), do systemu TTWCS, stworzonego specjalnie pod kątem systemu Tomahawk i umożliwiającego dodanie zupełnie nowych funkcjonalności, takich jak zmiana celu w czasie lotu pocisku.

Dodatkowym polem działania systemu Tomahawk, na którym nastąpił istotny postęp jest dołączenie w 1993 roku - a więc po 10 latach produkcyjnego użycia - systemu GPS, służącego do zdalnego namierzania lokalizacji celu. W tym wypadku system GPS częściowo zastąpił systemy TERCOM i DSMAC.

Systemy obrony powietrznej

Kolejną przykładową klasą systemów wielosystemowych są systemy antyrakietowe tworzone przez różne kraje w celu obrony własnej przestrzeni powietrznej. Wśród krajów, które wytworzyły własne wersje takiego systemu można wymienić: Stany Zjednoczone, Wielką Brytanię, Rosję, Francję, Chiny, Indie i Izrael (za http://en.wikipedia.org/wiki/Missile_defense).

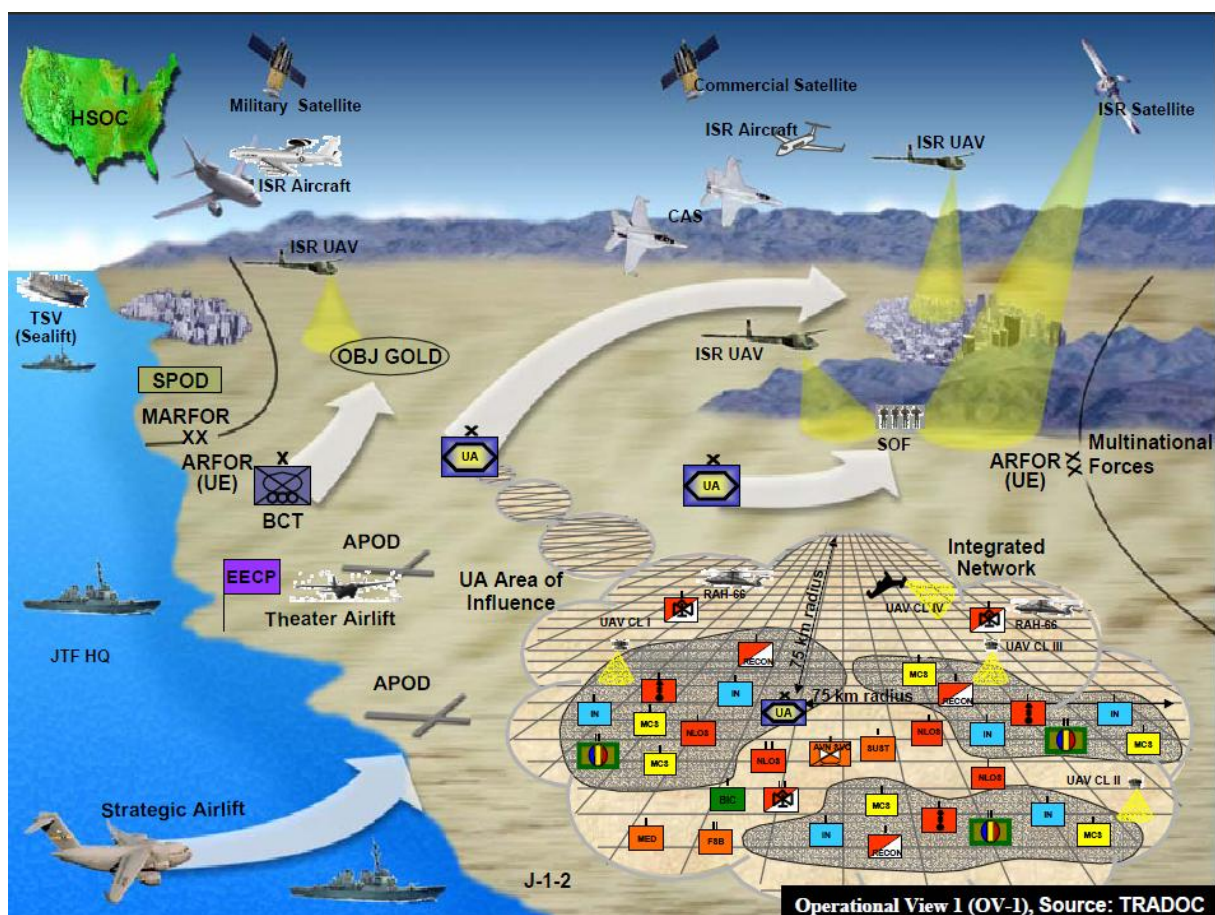


Rysunek 2 System obrony powietrznej Francji

Źródło: Bernard Homes „Governing testing systems of systems”

System nadzoru pola walki

System Task Force XXI to projekt Departamentu Obrony USA umożliwiający wspieranie żołnierzy na polu walki poprzez umożliwienie im skutecznej komunikacji na dużą odległość oraz dostarczanie informacji pochodzących z różnorodnych, normalnie niedostępnych dla pojedynczego żołnierza źródeł. Projekt pod różnymi nazwami jest realizowany od 1992 roku do tej pory, jego testowe wdrożenie miało miejsce w roku 1997, wnioski płynące z niego zostały opisane w pracy Anette Krygiel "Behind the Wizard's Curtain".



Rysunek 3 Task Force XXI

Źródło: Monica Farah-Stapleton „ Proposing a C4ISR Architecture Methodology for Homeland Security „

Systemy telekomunikacyjne

Ostatnią klasą systemów wielosystemowych są systemy telekomunikacyjne, wykorzystujące różnorodne metody przesyłania informacji na odległość do budowania nowych metod komunikacji i dostarczania funkcjonalności rozproszonym użytkownikom. Najczęściej przytaczanym przykładem systemu tego typu jest Internet, jednak trudno uznać go za system wzorcowy dla tej klasy chociażby ze względu na brak uporządkowanego systemu jego ewolucji. Inne przykłady systemów wielosystemowych z klasy systemów telekomunikacyjnych to systemy rezerwacji podróży, takie jak SABRE lub AMADEUS, lub zintegrowane systemy billingowe.

Cechy systemów wielosystemowych według Maier'a

Systemy wielosystemowe z definicji należą do zupełnie innej klasy złożoności niż systemy tradycyjne, jednak oprócz tej oczywistej właściwości można też wyróżnić kilka innych cech wspólnych dla tego typu systemów. Opisane w kolejnych punktach cechy są oparte na tzw. klasyfikacji Maier'a.

Pierwsze dwie są według niego warunkiem koniecznym zakwalifikowania danego systemu do klasy systemów wielosystemowych. Ostatnia z nich, niezależność geograficzna, miała istotne znaczenie w momencie powstawania pracy Maier'a (roku 1998), obecnie normą jest lokalizowanie komponentów jednego systemu w wielu niezależnych lokalizacjach.

Niezależność operacyjna

Oznacza ona, że poszczególne komponenty działają niezależnie od siebie, dostarczając części funkcjonalności systemu wielosystemowego. Dodatkowo mogą one pełnić oddzielną, często znacznie szerszą rolę jako niezależne systemy.

Niezależność zarządzania

Poszczególne systemy mogą być zarządzane poprzez niezależne organizacje, często konkurujące ze sobą lub realizujące niezależne cele. Wpływa to istotnie na cele stawiane systemowi wielosystemowemu oraz na sposób kontroli jego funkcjonowania.

Emergent behaviour („wyłaniające się” zachowanie)

Ze względu na złożoność systemów wielosystemowych trudno jest dokładnie przewidzieć i udokumentować ich zachowanie zanim docelowy system zacznie funkcjonować. Może to powodować konieczność wprowadzenia zmian w projekcie systemu w późniejszych etapach jego cyklu życia, co dodatkowo komplikuje proces wytwarzania systemu wielosystemowego.

Ewolucyjność rozwoju

Proces wytwarzania systemu wielosystemowego trudno uznać za ostatecznie zakończony, często jego poszczególne elementy muszą być dostosowywane do wymagań stawianych zarówno przez system wielosystemowy, jak i ich niezależną rolę. W takim wypadku istotna staje się możliwość weryfikacji poprawności nowej wersji komponentów np. poprzez testy regresji.

Niezależność geograficzna

Poszczególne komponenty systemu wielosystemowego są często zlokalizowane w różnych miejscach, a często także cechują się różnym sposobem i stopniem kontroli.

Cechy według CTAL

Złożone z wielu elementów i warstw

Na przykładzie wymienionych powyżej systemów łatwo zauważyć, że w przeciwieństwie do klasycznych systemów, które są projektowane i wytwarzane jako całość, systemy wielosystemowe stosunkowo rzadko posiadają strukturę hierarchiczną. Zazwyczaj są one w dużej mierze budowane z już istniejących komponentów, które muszą zostać dostosowane do współpracy w ramach wspólnej całości. Komponenty te mogą też należeć do różnych podmiotów, a także mogą znajdować się w różnych miejscach. Dodatkową komplikacją może być konieczność powielenia kluczowych funkcjonalności po to, by zapewnić ich redundancję na wypadek awarii.

Krytyczność dostarczanej funkcjonalności

Systemy wielosystemowe często tworzone są w celu dostarczenia funkcjonalności militarnej lub sprawowania kontroli nad ruchem pojazdów (samolotów, satelitów, rakiet itd.), co sprawia, że konieczne staje się spełnienie wymagań charakterystycznych dla systemów dostarczających krytycznej funkcjonalności (ang. life-critical).

Heterogeniczność komponentów

Systemy wielosystemowe mogą składać się z podsystemów o różnych typach budowy, a co za tym idzie różnej charakterystyce wytwarzania. Pierwszą „klasą” systemów będących komponentami

systemów wielosystemowych są systemy ładowalne. Obejmuje ona programy w pewnym stopniu niezależne od sprzętu na którym działają, co pozwala na łatwe i elastyczne zmiany oprogramowania. Zmiana sprzętu w przypadku systemów ładowalnych nie powoduje konieczności wymiany oprogramowania, jednak w takim przypadku powinna zostać przeprowadzona pewna forma testów regresji - ich brak może być tragiczny w skutkach, czego przykładem jest katastrofa rakiety Arienne 5 (więcej <http://goo.gl/Kt9ZP>).

Kolejnym typem systemów, wymagającym innego podejścia do wytwarzania i testowania to systemy wbudowane, obejmujące oprogramowanie ściśle związane z osprzętem, stanowiąca często jego część (np. firmware). Systemy tej klasy muszą spełniać ściśle wymagania stawiane przez osprzęt, będąc od niego wysoce zależne, co ogranicza możliwość zmian sprzętu i oprogramowania.

Ostatnią z podstawowych klas komponentów jest klasa systemów czasu rzeczywistego, zawierająca systemy, dla których wynik działania nie zależy tylko od poprawności osiągniętego rezultatu, ale także od czasu jego uzyskania. W przypadku przekroczenia tego czasu nawet prawidłowy - zgodny z przewidywaniami teoretycznymi - wynik jest uznawany za błąd.

Możliwość ewolucji komponentów

Oprócz kwestii ewolucyjności rozwoju istotnym dla działania systemu wielosystemowego jest fakt, że niektóre jego podsystemy funkcjonowały przed jego powstaniem, a więc ich dojrzałość i czas do zakończenia pracy może się diametralnie różnić od pozostałych jego części. Może to powodować konieczność zastąpienia ich innymi systemami, często różnymi w ramach systemu wielosystemowego i poza nim.

Różnice w stosunku do klasycznych systemów

W tej części pracy zostaną omówione główne różnice pomiędzy klasycznymi systemami a systemami wielosystemowymi.

Problemy organizacyjne

W przypadku systemów nadzorowanych przez organizacje o różnym stopniu rozwoju i sposobie zarządzania często dochodzi zarówno do niezgodności procesów towarzyszących wytwarzaniu oprogramowania, jak i do problemów podczas użytkowania systemu. Przykładowe polami konfliktu wymienianymi przez Sage'a są zarządzanie zmianą, zarządzanie ryzykiem i sposób współpracy zespołów obsługi i wsparcia systemu.

Problemy we współpracy

Oprócz problemów w samej organizacji pracy systemy wielosystemowe cechują też problemy we współpracy poszczególnych podmiotów je wytwarzających. Ich przykładowe źródła to współpraca państwowo – prywatna, utrudniona przez konieczność zachowania tajemnicy państwowej, rywalizacja między konkurencyjnymi firmami, powodująca niechęć do dzielenia się istotnymi informacjami, a także praktyki antymonopolowe sprawiające, że pewne zadania muszą zostać wykonane przez usługodawców niekoniecznie posiadających najwyższe kompetencje w określonej dziedzinie.

Konieczność zewnętrznej kwalifikacji

Systemy wielosystemowe zazwyczaj muszą spełnić narzucone zewnętrznie wymagania, związane z nadzorem państwowym – jak zbiór kryteriów Department of Defence Architecture Framework (DODAF) przypadku Departamentu Obrony USA – lub kwalifikacją organizacji odpowiedzialnych za ich nadzór, takich jak AWST w przypadku elektrowni atomowych, RTCA w przypadku systemów awiacyjnych czy CCHIT w przypadku systemów medycznych.

Zgodność z regulacjami jest sprawdzana zarówno poprzez weryfikację i walidację finalnego systemu, jak i sprawdzenie istnienia i poprawności dokumentacji stworzonej podczas procesu wytwarzania,

Wysoki stopień współpracy i zależności

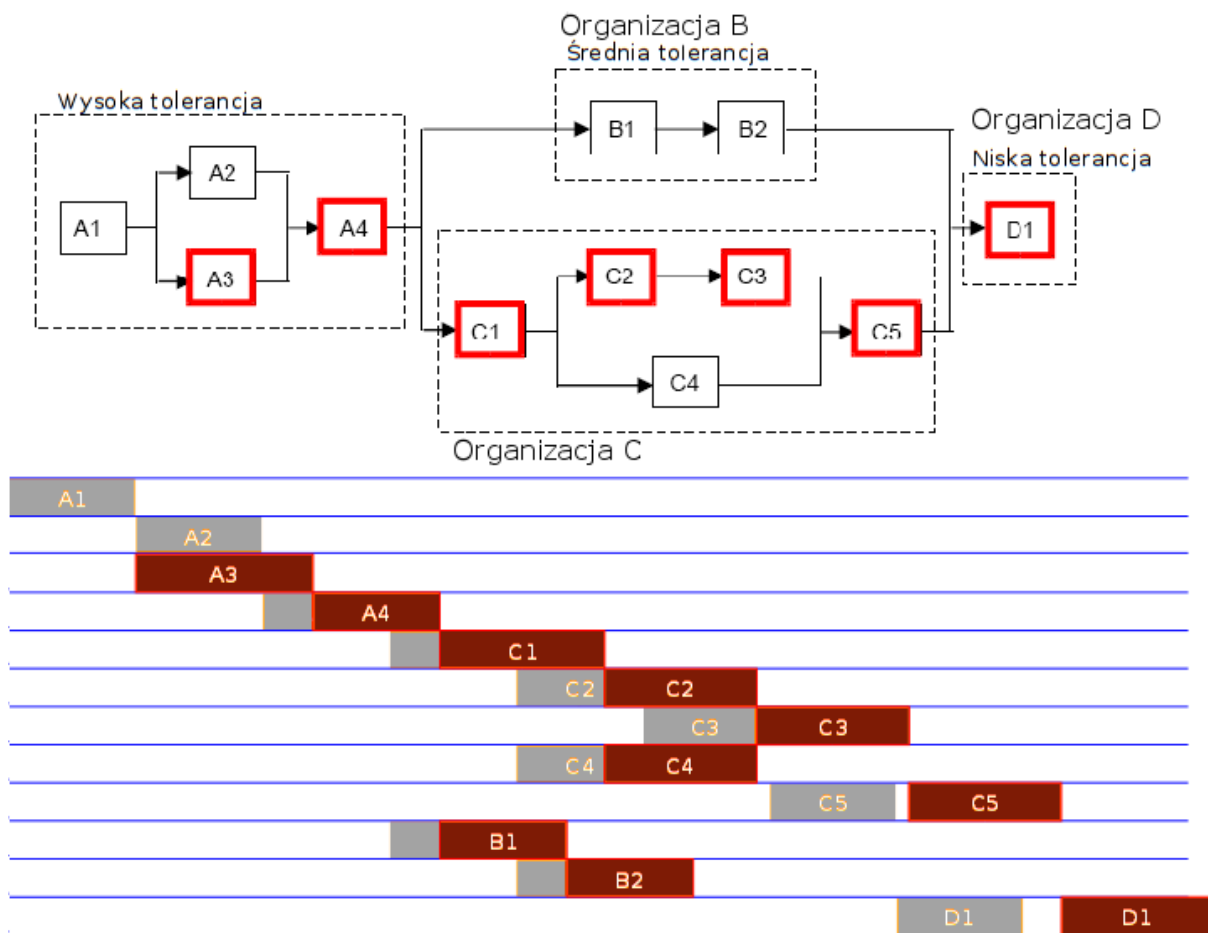
Biorąc pod uwagę duży stopień interakcji komponentów systemu wielosystemowego Bernard Homes zauważa dwa istotne problemy związane z zarządzaniem ryzykiem, które mają istotny wpływ na ich działanie. Pierwszym jest fakt, że w przypadku tak złożonych systemów ograniczenia wpływają nie tylko na bezpośrednio połączone systemy, ale także mają pośredni wpływ na funkcjonowanie całości systemu. Niezbędna jest zatem szczegółowa i nie ograniczająca się do jednego poziomu analiza ryzyka związanego z funkcjonowaniem i zmianami poszczególnych podsystemów. Drugą obserwacją jest to, że poszczególne systemy mogą mieć różną tolerancję ryzyka, często zależną także od ich właściciela/operatora. W tym przypadku niezbędna jest dodatkowa analiza uwarunkowań wpływających na ryzyko związane z poszczególnymi podsystemami.

Proponowanym przez Homes'a rozwiązaniem tych problemów jest rozpatrywanie nie tylko ryzyka związane z pojedynczym systemem, ale spojrzenie „wielopoziomowe”. W klasycznych systemach analiza ryzyka ogranicza się do rozpatrzenia zagrożeń związanych z działaniem systemu i wpływających na niego czynników zewnętrznych. W systemie wielosystemowym, biorąc pod uwagę wysoki stopień współpracy i zależności podsystemów równie istotna staje się analiza ryzyk przyjmowanych (ang. „inherited”) i powodowanych (ang. "imposed"). Dopiero wtedy można określić, z jakim ryzykiem wiąże się działanie określonego systemu i czy duplikacja jego funkcjonalności/zmiana sposobu współpracy nie pozwolą na zmniejszenie ogólnego poziomu ryzyka systemu.

Przykład

Kwestia tolerancji ryzyka w przypadku wytwarzania złożonych produktów w warunkach intensywnej współpracy może mieć istotny wpływ na ostateczny czas wytworzenia funkcjonalności, co obrazuje poniższy diagram stworzony przez Bernard'a Homes'a.

W tym wypadku poziom tolerancji ryzyka organizacji odnosi się do tolerancji na opóźnienia, zaś cały diagram ilustruje planowany i faktyczny przebieg wykonania określonego, złożonego zadania przez systemy należące do różnych organizacji. Jasnym kolorem oznaczone została planowana sekwencja wykonania poszczególnych zadań, zaś ciemnym faktyczna, wynikająca z opóźnienia spowodowanego w organizacji A.



Rysunek 4 Tolerancja ryzyka

Źródło: Bernard Homes „Governing testing systems of systems”

W tym przypadku względnie niewielkie opóźnienie, które wystąpiło w zadaniu A3 wykonywanym przez organizację A spowodowało bezpośrednio istotne opóźnienia w organizacji C, co ostatecznie doprowadziło do poważnego opóźnienia w wykonaniu całego zadania. Widać tutaj znaczenie zarówno wielopoziomowej analizy wpływów i ryzyka przyjmowanego i powodowanego - opóźnienie w organizacji D nie wynikało bezpośrednio z działania jej samej ani jej bezpośrednich sąsiadów. Istotna jest też kwestia zapewnienia odpowiedniej kolejności wykonania zadań i ewentualnej redundancji - gdyby działania organizacji D nie zależały od działań organizacji o większej tolerancji ryzyka lub gdyby jej wymagania mogły być dostarczone przez różne zadania poprzedzające można by uniknąć lub ograniczyć opóźnienie.

Wpływ złożoności systemu wielosystemowego na testowania

Specyfika systemów wielosystemowych sprawia, że proces ich testowania staje się nie tylko o wiele bardziej złożony, co w dość oczywisty sposób jest implikowane przez stopień skomplikowania samego systemu, ale także zmusza do rozwiązywania problemów nie występujące zazwyczaj w przypadku klasycznych systemów. Powoduje to nie tylko konieczność zaangażowania większej ilości środków, ale też konieczność zastosowania innego podejścia ze względu na fakt, że komponenty systemów wielosystemowych są niezależnymi systemami, z własnymi wymaganiami, ograniczeniami i

procesami życia. Podstawowe problemy w testowaniu systemów wielosystemowych wynikające z ich złożoności zostały opisane poniżej.

Ilość kodu

Złożoność systemu wielosystemowego to nie tylko suma złożoności jego komponentów, ale także złożoność interfejsów pomiędzy nimi, także tych umożliwiających redundancję. Kwestie integracji zostaną poruszone później, ale należy wziąć pod uwagę, że testowanie interfejsów między systemami jest zadaniem trudnym nawet w przypadku prostych systemów.

Wymaga ono kooperacji organizacji sprawujących kontrolę nad współpracującymi systemami, co z opisanych wcześniej względów może być problematyczne. Homes postuluje w takim przypadku rozdzielanie odpowiedzialności za testy w taki sposób, by systemy komponentowe były testowane przez zespoły odpowiedzialne za ich testowanie z ramienia właściciela, zaś interfejsy były testowane przez „obejmujący całość” (ang. overarching) zespół testerów systemu wielosystemowego, podlegając nadzorowi interesariuszy systemu wielosystemowego.

Procesy kontroli zmian

Przy wytwarzaniu klasycznych systemów proces kontroli zmian może być realizowany przy użyciu powszechnie dostępnych narzędzi do wersjonowania kodu, zarządzania zmianami oraz odpowiedniej dyscypliny. Heterogeniczność budowy systemów wielosystemowych (w tym obecność silnie powiązanego ze sprzętem oprogramowania wbudowanego) oraz problemy z współdzieleniem informacji i zarządzaniem prowadzą do poważnych komplikacji procesu kontroli zmian.

Dodatkowo w przypadku systemów wielosystemowych bardzo utrudnione jest precyzyjne planowanie procesu zmian, co wynika z możliwości niezależnej ewolucji komponentów systemu wielosystemowego i braku bezpośredniej kontroli nad ich wytwarzaniem.

Ilość i dostępność dokumentacji

Problemy związane z, będącą podstawą procesu testowego, dokumentacją systemów wielosystemowych można podzielić na dwie kategorie: związane z niedostatecznym udokumentowaniem systemu bądź brakiem dostępności istniejącej dokumentacji.

Pierwsza kategoria problemów powodowana jest przede wszystkim innowacyjnością stosowanych technologii i ich "wyłaniającym się" zachowaniem, które to cechy utrudniają umożliwiającej zaplanowanie i realizację testów. Przykładem takich cech mogą być na przykład problemy z oceną dokładności nowo powstających urządzeń naprowadzających w przypadku systemów kontroli pojazdów, czy też wydajność sprzętu budowanego na potrzeby konkretnego systemu telekomunikacyjnego.

Druga kategoria problemów związana jest z brakiem dostępności szczegółowej dokumentacji w przypadku gotowych i wcześniej istniejących systemów, których dokumentacja często nie jest udostępniana z przyczyn opisanych w poprzednim rozdziale.

Długi okres trwania projektu

Długi czas trwania projektu i różnice w etatach rozwojowych komponentów sprawiają, że w dłuższym terminie problematyczny staje się nadzór nad przetestowanymi funkcjonalnościami i spełnionymi wymaganiami. Systemy komponentowe, które w momencie projektowania systemu

wielosystemowego były wyznacznikiem standardów i wzorem nowoczesności w momencie jego wdrażania mogą być już przestarzałe. Może też zachodzić konieczność uzupełnienia ich o nowe funkcjonalności, które pierwotnie nie były uwzględnione w planie testów.

Dodatkowo w przypadku długich projektów występuje znacznie większa rotacja personalna, zarówno na stanowiskach wytwórczych jak i nadzorczych, co nie tylko wydłuża czas wytwarzania, ale też może być źródłem dodatkowych nieporozumień i konfliktów.

Inne kwestie związane z testowaniem systemów wielosystemowych

Opisane wcześniej problemy nie znaczą, że klasyczne metody testowania nie mają zastosowania dla systemów wielosystemowych. Nadal pełnią one fundamentalną rolę w procesie testowania komponentów, można też próbować zastosować ich ideę do testowania systemu wielosystemowego, tworząc niejako oddzielny proces testowy, działający na innym poziomie abstrakcji - pojedyncze systemy pełnią w nim rolę komponentów, a ich „testowanie” często polega na analizie wyników szczegółowego procesu testowego.

Testuj wcześniej i często

W przypadku systemów wielosystemowych stosowanie zasady „testuj wcześniej i często” jest zdecydowanie utrudnione. Podstawowymi problemami według Homes'a są:

- Problemy z dostępnością środowiska testowego
- Częste i trudne do przewidzenia zmiany konfiguracji
- Problem ewoluujących wymagań
- „Przepychanie” testów na wyższy poziom

Kwestia odpowiedzialności i nadzoru

W sytuacji, gdy własność i kontrola nad wytwarzaniem nie są skupione w rękach jednej organizacji, problemem staje się wyznaczenie osoby nadzorującej i odpowiedzialnej za przebieg projektu. Ze względu na to, że testerzy biorą - a na pewno powinni - brać czynny udział w wytwarzaniu systemu od jego najwcześniejszych etapów, nie reprezentując zazwyczaj interesów żadnej z pojedynczych organizacji są często postrzegani jako osoby, które odpowiadają za jakość systemu i kontrolę nad tym, w jaki sposób powstaje.

Zarządzanie ryzykiem

Jednym z istotnych wyników procesu testowania jest dostarczenie informacji na temat aktualnego poziomu ryzyka związanego z systemem. W przypadku systemów wielosystemowych jest to o tyle istotne, że poszczególne organizacje, a także pojedyncze systemy mogą mieć różną jego tolerancję. W tej sytuacji istotna staje się odpowiednia analiza zarówno ryzyka związanego z poszczególnymi systemami, jak i ryzyka powodowanego i przyjmowanego.

Klasyczną miarą ryzyka, opisującą wielkość potencjalnych strat można zaprezentować za pomocą formuły

*ryzyko = prawdopodobieństwo * koszt błędu*

Od poziomu ryzyka zależy to, na jakim szczeblu zarządzania dany błąd będzie obsługiwany. W przypadku systemów wielosystemowych takie podejście jest nie do końca wystarczające, np. Homes proponuje wyróżnienie trzech rodzajów „wpływu” (ang. impact) ryzyka na wyniki procesu:

- Koszt - mierzony procentem przekroczenia budżetu
- Funkcjonalność - określany przez to, jak poważny jest błąd
- Czas - procentowe opóźnienie lub ważność niedotrzymanych terminów

Zapewnienie spójności pomiarowej (ang. traceability)

Autorzy prac na temat systemów wielosystemowych zwracają uwagę na duże znaczenie zapewnienia spójności pomiarowej. Spójność pomiarowa jest rozumiana jako możliwość wzajemnego powiązania i przedstawienia historii oraz sposobu ewolucji wymagań, części kodu źródłowego, przypadków testowych i dostarczanych przez system funkcjonalności. Jest to szczególnie istotne w przypadku systemów wielosystemowych ze względu na opisane poniżej problemy.

W przypadku systemów wielosystemowych stworzenie środowiska testowego odzwierciedlającego środowisko produkcyjne jest zazwyczaj drogie i skomplikowane, a bywa także niemożliwe - chociażby ze względu na innowacyjność stosowanych technologii, implementowanych w pojedynczych prototypach. W tej sytuacji testy odbywają się po upływie długiego - w niektórych przypadkach mierzonego w latach - czasu od stworzenia wymagań, w czasie których sposób i cel testowania może się zmienić.

Ilość komponentów systemu wielosystemowego i stopień ich interakcji sprawia, że każda zmiana może mieć skutki w funkcjonowaniu nie tylko zmienianego komponentu, ale też tych współpracujących z nim bezpośrednio i pośrednio. W tym wypadku spójność pomiarowa służy zapewnieniu możliwości prześledzenia zmian, które mogą wpływać na daną funkcjonalność i zapewnieniu, że potencjalne odstępstwa od planu zostaną właściwie wykryte i obsłużone.

Biorąc pod uwagę, że testowanie systemu wielosystemowego przebiega na wielu poziomach, zarówno w odniesieniu do komponentów jak i całego systemu, istnieje pokusa działania mającego na celu zmianę organizacji odpowiedzialnej za przetestowanie problematycznej funkcjonalności. Jest to wygodne i uzasadnione z punktu widzenia pojedynczych interesariuszy, ale z punktu widzenia całego systemu opóźnia przeprowadzenie testów, a tym samym właściwą ocenę postępu projektu i ryzyka, co może powodować zwiększenie skutków ewentualnych błędów.

Brooks i Sage oraz Homes zwracają też uwagę na to, że w przypadku systemów systemowych bardzo istotną kwestią jest kontrola konfiguracji. Jest to o tyle istotne, że dotyczy nie tylko komponentów ładowalnych, a więc oprogramowania, ale też komponentów wbudowanych, w których oprogramowanie jest silnie połączone ze sprzętem na którym pracuje. W takim przypadku należy kontrolować nie tylko wersje poszczególnych systemów i ich ustawienia, ale też nawet sprzęt na którym działają. W przeciwnym wypadku przeprowadzenie wymaganej kwalifikacji systemu na podstawie raportów z testów może być niemożliwe.

Metoda CPA

Autorzy metody Cross Program Approach (CPA), Roland Brooks i Andrew Sage zakładają, że kluczem do skutecznego działania systemu wielosystemowego jest zapewnienie współdziałania (ang. interoperability) podsystemów będących jego komponentami.

Współdziałanie jest tu rozumiane jako zdolność systemu do poprawnej wymiany danych z innymi systemami poprzez wzajemnie wspierające zachowania (ang. mutually supportive actions), wymianę pożytecznych informacji i wiedzy w celu osiągnięcia prawidłowej wydajności operacyjnej. Współdziałanie nie oznacza przy tym, że interfejs użytkownika musi być identyczny dla wszystkich systemów, powinien raczej być dostosowany do konkretnego zadania i preferencji (język, kultura) jego operatorów.

Planowanie

Autorzy pracy sugerują użycie przy planowaniu metody Planowania opartego na Zdolnościach (ang. Capability Based Planning - CBP), skupiającej się na analizie obecnych i przyszłych braków w funkcjonalnościach planowanego systemu. Opiera się ona na podejściu top-down, a więc na przechodzeniu od ogólnych założeń wobec możliwości systemu do ich szczegółowej implementacji. Takie podejście umożliwia efektywną współpracę wielu organizacji, wykorzystując ich specyficzną wiedzę i doświadczenie do realizacji poszczególnych zadań.

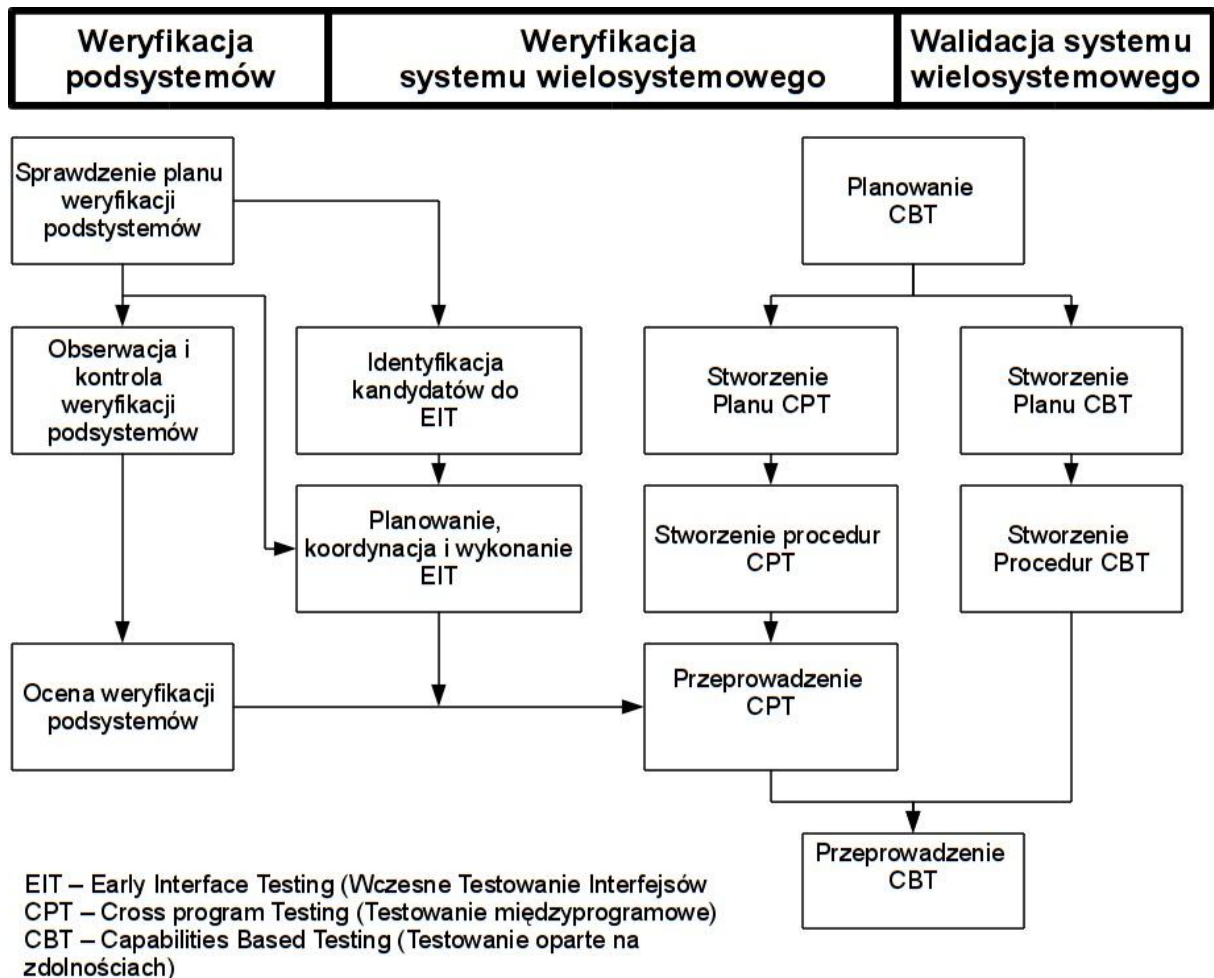
Specyfikacja wymagań

Tak jak zostało to opisane w rozdziale dotyczącym weryfikacji i walidacji, jednym z najistotniejszych wyzwań stojących przed projektantami systemu wielosystemowego jest właściwa analiza zdolności systemu i stworzenie odpowiednich wymagań. Brooks i Sage stwierdzają, że dobrze stworzone wymagania systemu wielosystemowego muszą:

- Umożliwiać wydajną i szybką implementację systemu w najczęściej stosowanych metodykach ewolucyjnych i spiralnych
- Specyfikować odpowiednie miary wydajności umożliwiające testowanie i weryfikację postępu projektu
- Zawierać minimalne ograniczenia (spełniane standardy, wymagania architektoniczne) i zachowania umożliwiające skuteczną integrację, współpracę i zarządzanie konfiguracją

Testowanie

Podstawą przeprowadzenia procesu testowania systemu wielosystemowego jest według Brooks'a i Sage'a Zintegrowany Plan Weryfikacji i Walidacji, którego struktura jest wyraźnie inspirowana wytycznymi normy IEEE 829. Proces testowania proponowany przez autorów metody CPA został przedstawiony na diagramie 5.1.



Rysunek 5 Proces testowania systemu wielosystemowego

Źródło: Opracowanie własne na podstawie Brooks, Sage *System of systems integration and test*

Brooks i Sage podzielili aktywności związane z testowaniem na cztery główne fazy, różniące się zarówno poziomem przeprowadzanych testów, jak i ich celem.

Faza testowania komponentów

Pierwszą z proponowanych faz jest faza testowania komponentów, obejmująca całościowy proces testowania podsystemów implementowanych od zera. W przypadku systemów, które są dostosowywane do działania w nowym środowisku powinny zostać przeprowadzone testy regresji, wskazana jest także weryfikacja kluczowych parametrów wydajnościowych już istniejących systemów.

Wczesne Testowanie Interfejsów

Jest to faza testów, której głównym zadaniem jest dostarczenie informacji o kompatybilności systemów komponentowych tak szybko, jak to możliwe. Nie musi ona przebiegać w sposób ściśle sformalizowany ze względu na brak powiązania z ostateczną kwalifikacją systemu, istotniejsza jest raczej szybkość identyfikacji potencjalnych zagrożeń i możliwość dostosowania do nich procesu wytwórczego.

Faza Wczesnego Testowania Interfejsów wymaga ścisłej współpracy zespołu odpowiadającego za testowanie systemu wielosystemowego z zespołami wytwórczymi. Oprócz możliwości kontrolowania postępu prac i poziomu realizacji wymagań taka kooperacja umożliwia też między innymi weryfikację poprawności projektu, wyspecyfikowanie środowiska testowego dla testów międzyprogramowych czy też znalezienie odpowiednich metod weryfikacji i walidacji poszczególnych zdolności.

Testy międzyprogramowe

Testy międzyprogramowe są formalną fazą testów, stanowiącą podstawę procesów weryfikacji i walidacji, a także kwalifikacji systemu wielosystemowego. Testy międzyprogramowe opierają się głównie na sprawdzeniu zgodności interfejsów z Dokumentami Kontroli Interfejsów oraz na zapewnieniu, że zdolności realizowane dzięki współpracy komponentów funkcjonują we właściwy sposób.

Każde testy rozpoczynane są od przeglądu gotowości do testów, który musi uwzględniać nie tylko stan oprogramowania podlegającego testowi, ale też zgodność architektury i konfiguracji z założonymi standardami. Testowaniu podlegają zarówno same funkcjonalności interfejsów, jak też ich zgodność z wyspecyfikowanymi wcześniej miarami wydajności.

Wynikiem testowania interfejsów jest nie tylko ocena jakości ich wykonania, ale też formalna decyzja o gotowości do integracji współpracujących systemów. W przypadku wystąpienia błędów konieczne jest stworzenie raportu niezgodności, który umożliwi ocenę związanego z nimi ryzyka i będzie podstawą oceny przyczyn i odpowiedzialności za wystąpienie niezgodności.

Testy oparte na zdolnościach

Jest to ostatnia faza testów systemu wielosystemowego, której głównym celem jest walidacja całości systemu wielosystemowego poprzez sprawdzenie jego poszczególnych zdolności. Polega ona na wykonaniu zaplanowanych wcześniej w procesie Planowania opartego na Zdolnościach scenariuszy sprawdzających implementację zdolności systemu w podejściu end-to-end. Oprócz wymagań funkcjonalnych sprawdzane są też wymagania niefunkcjonalne, takie jak użyteczność dla użytkownika końcowego, zarządzalność, wspieralność czy wydajność w środowisku produkcyjnym.

Metoda CTM

Pierwotnie model organizacji architektury obowiązujący w Departamencie Obrony USA był nazywany C4ISR, jednak po opublikowaniu jego drugiej wersji w 1997 kolejne wydanie, mające miejsce w roku 2003, nosiło już nazwę DODAF. Aktualnie dostępna jest już kolejna wersja DODAF'u oznaczona numerem 2.0. Podstawową metodą testowania systemów wielosystemowych używaną przy projektach realizowanych przy pomocy DODAF jest Metodologia Testów Zdolności (ang. Capability Test Methodology), która została opisana na podstawie oficjalnej dokumentacji. Zakłada ona podział procesu testowania na 6 faz.

Faza 1 - Stworzenie strategii testów i oceny

Faza pierwsza ma na celu wytworzenie dwóch głównych produktów: dokumentu strategii testów i Zintegrowanego Kontekstu Operacyjnego dla Testów (ang. Joint Operational Context for Test - JOC-T). Strategia testów definiuje sposób, w jaki zostanie określone to, czy system i jego zdolności spełniają postawione im wymagania, zaś JOC-T specyfikuje środowisko, w jakim system lub jego określona zdolność będzie funkcjonowała, a więc także będzie testowana.

Faza 2 - charakterystyka

Jest to faza, w której zarządcy projektu wraz z zespołem testowym uszczegóławiają strategię testów, identyfikując zdolności podlegające testowaniu, wymagane zasoby i tworząc harmonogram testów. Główne zadania w tej fazie to stworzenie określenie celu testowania, jego szczegółowych zadań i sposobu ich oceny oraz stworzenie dokumentu opisującego środowisko testowe oznaczane skrótem LVC-DE (ang. Live Virtual and Construction Distributed Environment).

Faza 3 - planowanie

W tej fazie na podstawie wytworzonych wcześniej dokumentów powstaje formalny plan testów. Pierwszym etapem tej fazy jest rozwój Projektu Testów (ang. Test Design) - polega na stworzeniu Planu Analizy Danych (ang. Data Analysis Plan) i Zintegrowanej Listy Wymagań (ang. Integrated Data Requirements List). Wytwarzanie DAP wymaga przeanalizowania produktów poprzednich faz w celu opracowania projektu testów poszczególnych zdolności. Podczas tego procesu analitycy powinni stworzyć i uaktualniać IDRL oraz dopracowywać szczegóły wymagań dotyczących zbieranych danych, które będą stanowiły podstawę dokumentu DCP (ang. Data Collection Plan - plan gromadzenia danych). Ostatecznie dokument DAP powinien specyfikować metody i procesy niezbędne do zbierania danych testowych i dostarczania na ich podstawie ilościowych i jakościowych wyników testów. Drugim etapem fazy planowania jest przeprowadzenie analizy wymagań i stworzenie dokumentacji funkcjonalnej dla środowiska testowego LVC-DE. Trzecim, końcowym etapem jest stworzenie planu testów zawierającego:

- Dane zgromadzone w DAP
- Spis parametrów funkcjonalnych
- Projekt funkcjonalny LVC-DP
- Analizę procesów wspierających testowanie
- Harmonogram testów

Faza 4 - Stworzenie środowiska testowego

Faza czwarta zakłada stworzenie środowiska testowego LVC-DE składającego się z komponentów produkcyjnych (**Live**), stworzonych specjalnie na potrzeby testów (**Construction**) i wirtualnych (**Virtual**), umożliwiających testowanie w środowisku odzwierciedlającym warunki operacyjne systemu wielosystemowego, a więc rozproszonym (**Distributed Environment**).

Pierwszą z czynności w tej fazie jest stworzenie projektu LVC-DE, specyfikującego wymagania wobec środowiska testowego, które umożliwi przetestowanie systemu z uwzględnieniem parametrów funkcjonalnych zawartych w planie testów. Drugą z nich jest fizyczne wytworzenie komponentów środowiska testowego, z użyciem zweryfikowanego i zwalidowanego projektu LVC-DE. Trzecią z nich jest integracja komponentów środowiska testowego (sprzętu, oprogramowania, baz danych i sieci) i przetestowania ich w celu sprawdzenia, czy spełniają założone wymogi i są zgodne z kryteriami wejścia procesu testowania.

Faza 5 - Zarządzanie wykonaniem testów

W tej fazie zespół odpowiadający za testowanie systemu wielosystemowego powinien skupić się na analizie postępów procesu testowego realizowanego przez zespoły testowe poszczególnych wytwórców. Głównymi zadaniami są tu kontrola raportów testów, reagowanie na nieprzewidziane zdarzenia i zapewnienie, że poszczególne zespoły pracują zgodnie z planem i (tam gdzie to potrzebne) w sposób zsynchronizowany. Ważne jest też zapewnienie, żeby kolejne aktywności testowe rozpoczęły się dopiero po zakończeniu się poprzedzających działań i ich ocenie według wcześniej zdefiniowanych kryteriów.

Faza 6 - Ocena zdolności systemu

Jest to ostatnia faza testowania, która jest podstawą procesu weryfikacji i walidacji. Polega ona na analizie zgromadzonych danych, wyników testów i zademonstrowanych zdolności w celu ocenienia wydajności systemu wielosystemowego i jego pojedynczych funkcjonalności oraz zależności między zaobserwowaną wydajnością a miarami efektywności działania. Faza ta często ma przebieg iteracyjny, poszczególne analizy są powtarzane w zależności od tego, jaka część systemu wielosystemowego / cecha / zadanie jest aktualnie rozpatrywane. Pierwszym etapem oceny zdolności systemu jest analiza danych. Polega ona na przekształceniu danych testowych na informację o tym, co i dlaczego wydarzyło się podczas testów.

Dane ilościowe i jakościowe zebrane podczas testowania umożliwiają porównanie efektywności działania testowanego systemu z miarami wydajności dotyczącymi atrybutów i zadań systemu wielosystemowego. Dane zgromadzone podczas różnych etapów i typów testów są także analizowane statystycznie w celu potwierdzenia istotności wpływu testowanych funkcjonalności na realizowane przez nie zdolności systemu wielosystemowego, a także w poszukiwaniu nowych wyników, które wystąpiły podczas testowania.

Drugim etapem oceny zdolności systemu jest wykorzystanie wyników analizy danych do oceny, czy i jak dobrze testowany system spełnia stawiane mu wymagania. Jest to podstawą formalnych rekomendacji dostarczanych komitetowi sterującemu przez zespół testowy, a w konsekwencji decyzji dotyczących dalszego przebiegu procesu wytwarzania.

Case'y

W tym rozdziale zostaną zaprezentowane prace, które zawierają praktyczne informacje na temat testowania systemów wielosystemowych.

Krygiel - „Behind the Wizard's Curtain”

Książka "Behind the Wizard's Curtain" zawiera opis wytwarzania, integracji i testowania dwóch systemów:

- Task Force XXI - zintegrowanego systemu zarządzania jednostkami piechoty, wykorzystującego różnorodne formy komunikacji i kontroli pola walki
- Digital Production System - systemu, który miał za zadanie digitalizację materiałów GIS (Geospatial Information and Services - Informacji i Serwisów Geoprzestrzennych) zgromadzonych przez największą agencję kartograficzną świata, Defense Mapping Agency (obecnie National Geospatial-Intelligence Agency).

Oba systemy były wytwarzane według wytycznych Departamentu Obrony, a więc z użyciem metodyki C4ISR. Krygiel w swej pracy skupia się na procesie integracji systemu wielosystemowego, jednak we wnioskach przedstawia też kilka zaleceń dotyczących testowania.

Pierwszym z nich jest sformułowanie i zaakceptowanie przez wszystkie zainteresowane strony planu i procesu testów, który uwzględnia zidentyfikowane i uzgodnione podczas projektowania i wytwarzania ryzyka. Jest to podstawą do przeprowadzenia testów i oceny znaczenia ich wyników. Biorąc pod uwagę, że oba systemy były testowane w dużej mierze przez ich operatorów, reprezentujących różne organizacje o niekoniecznie zgodnych celach istotne jest wcześniejsze uzgodnienie i akceptacja scenariuszy testowych i sposobu interpretacji ich wyników.

Drugim zaleceniem jest dobór scenariuszy testowych w taki sposób, by były one reprezentatywne dla wymagań funkcjonalnych stawianych przed produkcyjnym systemem. Zalecenie takie nie jest szczególnie zaskakujące, jednak zwłaszcza w przypadku systemu Task Force XXI, którego testy polegały na wykonaniu ćwiczeń na polu walki i scenariuszy bojowych dobór testowanych zagadnień stanowił duży problem. Po zakończeniu testowania i integracji w przypadku obu systemów pojawiły się problemy dotyczące zagadnień, które zostały uznane za niewymagające testowania.

Trzecim zaleceniem Krygiel jest użycie w procesie testowania produkcyjnego systemu wielosystemowego lub jego wierniej kopii, tak jak miało to miejsce w przypadku DPS. Jest to o tyle istotne, że oba testowane systemy (w przeciwieństwie np. do systemów kontroli pocisków / pojazdów) charakteryzowały się dużą intensywnością pracy operatorów. W takim wypadku ważne jest, by operatorzy mieli do czynienia z taką wersją systemu / interfejsu, jaka będzie używana produkcyjnie. Krygiel zauważa też, że symulowanie niedostępnych komponentów (zazwyczaj za pomocą uproszczonych modeli) oprócz oczywistego problemu jakości odwzorowania wprowadzało też dodatkowe problemy związane bezpośrednio z funkcjonowaniem symulatorów, co dodatkowo utrudniało testowanie.

We wnioskach z testowania nie została bezpośrednio zawarta informacja, że wiele interfejsów była testowana w sposób nieformalny długo przed rozpoczęciem fazy integracji. W opisie procesu wytwarzania Krygiel podkreśla jednak, że miało to pozytywny wpływ na proces wytwarzania systemu. Widoczne jest tu podobieństwo do fazy Wczesnego Testowania Interfejsów opisanej w metodzie Cross Program Approach.

Sledge - „Army ASSIP System-of-Systems Test Metrics Task”

Kolejną pracą dotyczącą tematu testowania systemów wielosystemowych jest publikacja Carol H. Sledge z 2006 roku. Dotyczy ona metryk używanych w procesie integracji i testowania systemów wielosystemowych, ale zawiera też szereg rekomendacji dotyczących tych procesów. Praca powstała na podstawie analizy sposobu powstawania i współpracy z wytwórcami i zarządzającymi systemem Warfighter. Te, które można odnieść do ogólnego procesu testowania systemów wielosystemowych zostały opisane poniżej.

Sledge sugeruje, że w celu właściwego nadzorowania postępów pracy nad systemem wielosystemowym należy ustanowić „obejmujący całość” zespół zajmujący się kierowaniem systemem wielosystemowym (ang. overarching SoS executive). Ma to spowodować stworzenie jasnej wizji systemu wielosystemowego jako całości, co ułatwia podejmowanie decyzji właściwych z punktu widzenia pełnego projektu oraz umożliwia właściwy podział, a co za tym idzie finansowanie

podejmowanych działań. Stworzenie takiego kierownictwa może pozwolić uniknąć opisanych wcześniej problemów organizacyjnych, charakterystycznych dla systemów wielosystemowych. Ułatwia też zachowanie niezależności zespołu odpowiedzialnego za testowanie całości systemu wielosystemowego.

W drugiej rekomendacji Sledge podkreśla znaczenie właściwego zdefiniowania zdolności systemu wielosystemowego oraz stworzenia formalnego planu testów umożliwiającego ich weryfikację. Autorka podkreśla przy tym konieczność większego niż wcześniej (w odniesieniu do systemu Warfighter XXI) udziału użytkowników końcowych w procesie definiowania zdolności, które powinny zostać zawarte w Dokumencie Wstępnych Zdolności (ang. Initial Capabilities Document - ICD).

Sledge postuluje też stworzenie na podstawie ICD formalnego planu testów systemu wielosystemowego, którego brak skutkowało według niej:

- Brakiem zrozumienia sposobu testowania poszczególnych wymagań przez ogół odpowiedzialnych za testowanie
- Problemami w stworzeniu środowiska testowego odpowiedniego do weryfikacji i walidacji wymagań, a przy tym odzwierciedlającego środowisko produkcyjne
- Testowaniem "tyle ile zdążymy", czyli dopóki nie nadejdzie deadline, zamiast „aż produkt będzie wystarczająco dobry"
- Brakiem zrozumienia pojęcia "wystarczająco dobry", a co za tym idzie kryteriów gotowości systemu do wdrożenia

Trzecia rekomendacja podkreśla znaczenie działań mających na celu integrację i testowanie nowo powstających systemów / interfejsów jeszcze przed rozpoczęciem właściwej fazy testów systemu wielosystemowego (jak w metodzie CTM). Umożliwia to weryfikację spełnienia przez poszczególne komponenty często zmieniających się wymagań wobec ich roli w systemie wielosystemowym, dostarcza informacji na temat poziomu ryzyka, a weryfikacja protokołów z nieformalnej fazy testów może stanowić formę "testów dymnych" (ang. smoke test), kwalifikujących dany produkt do rozpoczęcia formalnego testowania. Takie podejście umożliwia zarówno redukcję / właściwą alokację kosztów procesu testowego, jak i skrócenie czasu potrzebnego na naprawę ewentualnych usterek.

Czwartą rekomendacją Sledge jest stworzenie centralnego systemu zarządzania defektami, wspólnego zarówno dla całości systemu wielosystemowego jak i poszczególnych podsystemów. Oprócz pełnej kontroli nad ilością i poważnością defektów umożliwia to właściwe ustalenie, wyliczanie i modyfikowanie metryk i miar opisujących proces testowania. Stworzenie takiego systemu jest powszechną praktyką w wytwarzaniu oprogramowania, jednak według Sledge doświadczenia zdobyte podczas wytwarzania systemu Warfighter XXI wskazują na dwa pola, na których powinna nastąpić poprawa.

Po pierwsze, większość defektów była zgłaszana w odniesieniu do systemów komponentowych, jednak bez właściwego powiązania ich z wymaganiami w stosunku do całości systemu wielosystemowego. W tej sytuacji zarządzający całością procesu wytwarzania mieli dużo gorszy przegląd sytuacji, musząc samemu analizować skutki defektów dla systemu wielosystemowego.

Po drugie, defekty powinny obowiązkowo zawierać informacje dotyczące sposobu kontaktu z osobami i organizacjami do których są przypisane i których mogą dotyczyć, co powinno umożliwić efektywną komunikację.

Połączenie sugestii zawartych w rekomendacji 3 i 4 umożliwia stworzenie i egzekwowanie minimalnych wymagań wobec każdego systemu dostarczanego do formalnego środowiska integracji i testów. Negatywnymi skutkami braku takich wymagań są:

- Brak możliwości "stabilizacji" testowanego produktu, a co za tym idzie precyzyjnego określenia wersji testowanych komponentów
- Duży wysiłek poświęcony na instalację i testy regresji poprawek (przy dostarczaniu ich na bieżąco)
- Ogólne zmniejszenie wydajności zespołu testowego i innych osób biorących udział w testach
- Straty ponoszone przez wszystkie organizacje, nawet jeśli zawiniła tylko jedna z nich

Ostatnia rekomendacja dotyczy sposobu prezentacji wyników otrzymanych podczas testowania systemu. Sledge zaleca przy tym wyliczanie zarówno metryk postępu testów, jak i metryk jakości (ang. goodness) przetestowanej części systemu. Dane te powinny być podstawą ewentualnych zmian w procesie testowym, np. decyzji o rozmiarze wątków testowych czy ograniczeniu / rozszerzeniu zakresu testowania określonej funkcjonalności.

Brooks i Sage - „System of systems integration and test”

Brooks i Sage w swojej pracy oprócz metody testowania Cross Program Approach prezentują też przypadek jej zastosowania do implementacji i testowania systemu wielosystemowego (którego nazwa nie została jawnie podana). Był on wytwarzany w metodyce spiralnej przez niezależnych podwykonawców, zaś integracją i testowaniem na zlecenie Departamentu Obrony USA zajmowała się firma Lockheed Martin, nie powiązana z żadnym z dostawców.

Ze względu na skalę i specyfikę systemu większość projektów implementacyjnych musiało spełnić wymagania formalne specyficzne dla projektów klasy Major Defense Acquisition Programs, co stanowiło dodatkowe wyzwanie dla procesu testowania. Autorzy pracy przy analizie skuteczności metody CPA używali framework'u zaproponowanego w pracy *Case Studies of Systems Engineering and Management in Systems Acquisition* autorstwa Friedman'a i Sage'a. Umożliwił on nie tylko spójną i efektywną interpretację zgromadzonych danych, ale też właściwą analizę podziału obowiązków między trzy główne domeny odpowiedzialności:

- Odpowiedzialność wytwórców
- Odpowiedzialność głównego integratora (Lockheed Martin) i właściciela (Departamentu Obrony)
- Odpowiedzialność właściciela

Brooks i Sage przedstawili w swojej pracy 8 sugestii dotyczących testowania i integracji systemów wielosystemowych, podążając za podziałem zawartym w używanym przez nich framework'u. Płynące z nich wnioski można podzielić na trzy główne kategorie.

Po pierwsze, bardzo istotne jest jak najwcześniejsze zaangażowanie w pracę osób odpowiedzialne za integrację i testowanie. Z doświadczeń zawartych w pracy wynika, że tak naprawdę powinny brać udział w projektowaniu systemu od pierwszych jego etapów, czyli analizy wymagań i zdolności systemu. W opisywanym przypadku w pierwotnie stworzonych wymaganiach wykryto wiele braków, niespójności i powtórzeń, które mogły być przyczyną poważnych dodatkowych kosztów i opóźnień. Zostały one wykryte przez Głównego Integratora Systemu (ang. Lead System Integrator), firmę Lockheed Martin, i także przez nią naprawione.

Po drugie, ważne jest, aby wszelkie kluczowe decyzje dotyczące projektu były podejmowane biorąc pod uwagę całość systemu wielosystemowego. Naturalnym jest, że przedstawiciele poszczególnych zespołów zwracają uwagę przede wszystkim na kwestie dotyczące ich dziedziny, pomijając kwestie nie dotyczące ich bezpośrednio. W przypadku systemów wielosystemowych jest to o tyle bardziej znaczące, że szczegółowa znajomość całego systemu jest praktycznie niemożliwa, a najistotniejsze funkcjonalności są dostarczane przez komponenty wytwarzane przez różnych dostawców. W celu ułatwienia podejmowania właściwych decyzji Brooks i Sage zalecają stworzenie modelu architektonicznego umożliwiającego całościowe spojrzenie na wytwarzany system, z użyciem powszechnie przyjętych i zrozumiałych modeli komponentów. Po trzecie, Brooks i Sage kładą duży nacisk na planowanie działań związanych z testowaniem i integracją oraz procesów je wspierających.

W kwestii weryfikacji i walidacji systemu autorzy podkreślają (zgodnie z przedstawioną przez nich metodą CTA) znaczenie stworzenia Zintegrowanego Planu Weryfikacji i Walidacji, będącego podstawą procesu testowego. Dzięki ustaleniu wspólnego dla wszystkich programów wytwórczych okresu trwania cyklu wytwarzania możliwe było jednoczesne testowanie najnowszej wersji poszczególnych komponentów. Istotne okazały się też wnioski wyciągnięte z fazy Wczesnego Testowania Interfejsów, które umożliwiły efektywne zarządzanie ryzykiem i zaplanowanie zarówno fazy testów przedwdrożeniowych, jak i dokonywanych w czasie funkcjonowania systemu.

Joglekar - "Test Strategy for Net-Centric C4ISR System"

Anil N. Joglekar porusza w swojej pracy kwestię systemów C4ISR opartych na sieciach (ang. Net-Centric C4ISR Systems). Przewiduje on, że w przyszłości coraz więcej systemów wielosystemowych będzie powstawało dzięki połączeniu funkcjonalności już istniejących systemów w celu uzyskania nowej, wspólnej funkcjonalności, w przeciwieństwie do obecnego trendu budowania nowych aplikacji dla nowych zadań.

Podkreśla on tym samym istotność właściwych testów integracyjnych i regresji, koniecznych przy łączeniu systemów nie projektowanych do określonego typu współpracy. Joglekar zauważa też, że większość systemów wielosystemowych jest sterowana przez technologię (ang. technology driven).

Skutkuje to ich szybkim starzeniem (ang. obsolescence), które rodzi presję na przyspieszanie tempa wytwarzania i wdrażania produktu, ograniczając tym samym czas i środki dostępne na testowanie. W tej sytuacji kluczowa staje się decyzja, co i w jakim stopniu (a także jakiej kolejności) zostanie przetestowane. Dodatkowo, jak zostało to wspomniane w rozdziale dotyczącym specyfiki systemów wielosystemowych w przypadku nowoczesnych, często powstałych na potrzeby konkretnego projektu technologii możliwość i jakość ich symulacji na potrzeby procesu testowego jest dyskusyjna.

W tej sytuacji często jedynym wyjściem staje się wykorzystanie w testach części środowiska produkcyjnego. Joglekar proponuje odseparowanie komunikacji na potrzeby testów za pomocą

oddzielnego szyfrowania, co według niego (przy zachowaniu właściwej przepustowości) powinno umożliwić realistyczne testy bez negatywnego wpływu na już funkcjonujące systemy.

Joglekar sugeruje stosowanie kilku zasad, które powinny umożliwić właściwe testowanie systemów wielosystemowych. Pierwszą z nich jest wykonywanie testów częściowo w środowisku produkcyjnym i przykładanie dużej wagi do sprawdzania założeń całego systemu wielosystemowego. W rozpatrywanym przez niego przypadku systemów pola walki jest to możliwe dzięki wykorzystaniu odmiennego sposobu szyfrowania wiadomości, co pozwala na testowanie nowych funkcjonalności w ich docelowym środowisku bez konieczności jego duplikowania.

Weryfikacja spełnienia założeń jest możliwa także dzięki temu, że w ostatecznych testach biorą udział jego użytkownicy - w przypadku rozpatrywanym przez Joglekar'a żołnierze i operatorzy systemów pola walki. Jest to możliwe dzięki temu, że do celów testowych wykorzystywane są także realne sposoby wydawania rozkazów i sterowania systemami komponentowymi. Zalecenie to jest odzwierciedleniem znanej z systemów niemilitarnych fazy testów akceptacyjnych wykonywanych przez użytkowników końcowych, nazywanej w literaturze angielskojęzycznej UAT - User Acceptance Testing.

Kolejnym zaleceniem, odzwierciedlającym podejście opisane w Cross Program Approach, jest stworzenie zespołów odpowiedzialnych za wczesne/ciągłe testowanie powstających komponentów. Joglekar zaleca przy tym, by zapewnić niezależność testerów całości systemu wielosystemowego, ale jednocześnie umożliwiać ich współpracę z poszczególnymi zespołami wytwórczymi.

Ostatnim zaleceniem jest stosowanie metody testowania opartego na ryzyku do testowania poszczególnych produktów pośrednich procesu wytwarzania oraz weryfikowanie poprawności działania procesu kontroli i dowodzenia (ang. command and control) za pomocą ustalonych wcześniej scenariuszy.

Podsumowanie

Zagadnienie systemów wielosystemowych przedstawione zostało przedstawione w artykule w sposób skrótowy, powinno jednak dawać ogólne pojęcie o specyfice tych systemów i trudnościach, jakie występują przy ich testowaniu. Oprócz dwóch przedstawionych metod – CPA i CTM – w pracach dotyczących systemów wielosystemowych często wspomniana jest metoda Testowania Opartego na Ryzyku, opisana szczegółowo w 4 numerze magazynu Core.

Niewątpliwie możliwość praktycznej weryfikacji znacznie podniosłaby wartość opisu metod testowania systemów wielosystemowych, jednak ze względu na specyfikę systemów wielosystemowych trudno mówić o "symulacji" procesu ich wytwarzania na potrzeby artykułu czy konsultacji wniosków z osobami posiadającymi odpowiednie doświadczenie. W przyszłości systemy wielosystemowe będą się stawały coraz bardziej popularne, co powinno spowodować zwiększenie dostępności informacji i ocenę, czy opisywane podejście jest właściwe.

Referencje

Roland T. Brooks, Andrew P. Sage. System of systems integration and test, 2006.

Monica Farah-Stapleton. Proposing a C4ISR Architecture Methodology for Homeland Security, 2004.

Bernard Homes. Governing Testing of Systems of Systems. TestWarez Krakow, 2009.

ISTQB. Certified Tester Advanced Level Syllabus. <http://www.istqb.org/download.htm>, 2007.

Annette Krygiel. Behind the Wizard's Curtain. NDU, 1999.

Mark W. Maier. Architecting Principles for Systems-of-Systems, 1998.

Jeffrey S. Mayer. Tactical Tomahawk Weapon System Developmental/ Operational Testing Testing a System of Systems., 2005.

Andrew P. Sage, Christopher D. Cuppan. On the Systems Engineering and Management of Systems of Systems and Federations of Systems, 2001.

Carol A. Sledge. Army ASSIP System-of-Systems Test Metrics Task. <http://www.sei.cmu.edu/reports/06sr011.pdf>, 2006.