Business Use Case: Activity Streams delivering content for required action based
on automatically triggered events.

1. Use Case Identification -

Created By:	IBM
Date Created:	5/8/2012
Version	1.1

1.

2. Actors

Primary Actor: Software Developer
Secondary Actor: Software Tester

3. Definitions

Software Developer – A person responsible for creating a software product. Runs continuous integration builds to generate a software package that can be installed or deployed onto a standalone computer or a cloud computing resource.
Software Tester – A person responsible for verifying the functionality of a software product.
Source Control/Build System – A software system that supports a software repository capability and a compilation/build mechanism that packages software for installation.
Test System – A software system capable of deploying, executing and collected automated test results. The test system can be a separate system known as a test harness, or can be a set of software programs built and installed with the product build. The actual form a system takes is not important for this use case.
ActivityStream – An ActivityStream is a list of recent activities generated around a particular work task or artifact or process. Each activity in an ActivityStream is represented in a standard JSON (or any mark up language) format.
OpenSocial Gadget/Embedded Experience – A user interface component that provides a small, granular view of a particular aspect of an application. The gadget can be embedded within an activity in an ActivityStream to create an embedded experience.

4. Background & Business Goals

In software development, continuous integration seeks to minimize the amount of changes that go into a build so that small, incremental changes to the software package can be adequately tested. By using automated deployment of the software package into a cloud environment, the amount of time required to consistently provision and configure a test environment is significantly reduced. An agile development team can publish upwards of 50 builds a day.
Specific members of the test team need to be notified when a test environment is ready for additional testing. They need information regarding the build content (such as which development work items were included), the quality of the build (such as whether the automated test cases have passed), the location of the formal test environment (such as a particular cloud instance), and which additional test scripts need to be executed by the tester. When the tester completes the acceptance testing, the verified build can be promoted into a live production environment.
The test activities for a tester are not optional, but a required part of the deployment process. Verification activities for a build are required. For builds containing high priority fixes, such as security patches, verification of the build must take priority over other tasks.

5. Concept of Operations

5.1. Current Service or Solution

There are two common ways a development or test team member remain aware of the build status. First, they can watch a build progress view or webpage. Unlike developers, testers do not initiate builds, so a build progress view provides little value to them within their existing tool sets.
Second, a person can subscribe to e-mail or instant message notifications on the build system. These message-based notifications generally require two application context switches – one: from the tester's tool to the e-mail tool, and two: a launch-in-context from the e-mail tool to the build tool. The messaging application must be running on the tester's machine to get the initial notification, and the tester might need to do an additional login to access data on the build system. Information about the build, such as automated test status and the list of fixes in the build may require additional digging in order to assemble the information.
Conveying the urgency/priority of verifying a particular build is frequently done via verbal or instant messaging communications in separate tools. Tracking the completion of verification tasks is frequently done in separate tools that do not link back to the build or development environments.

5.2. Desired Social Business Implementation

Build and automation notifications are sent to developers and testers through an Activity Stream that can be viewed in context to their preferred work tool. Developers are notified whether the build succeeded or failed. Required testers are notified only if they have testing responsibilities specific to that build and only if the build succeeded. They also receive information on the list of changes in the build as well as the priority of the fixes in the build.
Other interested testers may get notified if they expressed an interest (for example, if they subscribed to all build events or if they are following a particular work item). By using an Activity Stream located in their application of choice, the notifications can be targeted directly and contextually to the interested parties with an embedded experience that enables them to further interact with the build results.
The access to the activities should be standardized to facilitate the integration of a heterogeneous tool set that is frequently encountered around the development/build/test echo system.

5.2.1 Benefits

Instant Messaging solutions do not lend themselves to this particular scenario because the activity sent to the tester's ActivityStream is automated. The test automation system generates the activity when the test run is complete. The use of instant messaging implies manual intervention and makes the inclusion of contextual data about how the automated tests ran difficult to include.
Events could be published to a tester using other event technology such as RSS, Atom or even MQTT. However, the ActivityStream format is currently the best-fit format for distributing activities that contain embedded experience gadgets, which are fully functional read/write mini-applications in context to the activity.
Atom publishing technology, by itself, lacks the ability to express much of the activity-specific metadata in a machine-parsed format for user activities. With extensions specified in http://activitystrea.ms/specs/atom/1.0/, Atom technology can be used to publish events to a stream.
The rich metadata present in an ActivityStream-compliant stream makes more data available and a richer ecosystem of supporting software exists that could take advantage of it.
ActivityStream information can be stored and processed via analytics if necessary for additional insights into activities and solutions taking place in the organization.

Prioritization of tasks can be handled via ActivityStreams (with a specification modification). Tracking of required tasks such as verification of a build can be fed to any number of systems from capabilities provided by Embedded Experiences in the testers ActivityStream, saving them time and effort of switching contexts and tools.

6. Social Business Standards

ActivityStreams
OpenSocial Gadgets / Embedded Experiences
OAuth 2

6.1. Standards Gaps

For ActivityStreams:
There needs to be a way to articulate the **priority** of activities. This can manifest itself in a UI in a variety of ways, but the information needs to be contained in the ActivityStream message itself in order to be displayed correctly.
There needs to be a way to indicate that a task is **required** rather than optional. This is particularly true if an ActivityStream contains a significant amount of information, some of which is informational and some of which indicates required actions that must be taken.

6.2. Discussion

The priority of a task can be manifested in a variety of ways on a UI. However, this particular discussion focuses on priority as set in an activity using the activity stream standard. The priority field is initially set by the sender of the activity and is based on the sender's (automated or manual) desire to have a particular activity be deemed more important than others. The activity stream service handling the activity can adjust priorities and, as mentioned earlier, UI visualization and user settings can address or ignore priorities as they see fit. Currently, there is no way to even specify a priority and prioritization is often done by examining other fields in an activity such as the sender or the target or by examining specific attributes or content in the activity.
A required activity is one that must be viewed and responded to by a particular target of the activity. Requiring a response or action from an activity in an activity stream is problematic because it implies an embedded experience with certain capabilities must be included with the activity to allow a response. There is further opinion that there is a difference between an activity and activity stream where events are emphemeral as compared to as task list where work items must remain until completed. The activity stream standard currently provides a few extra attributes that mark an activity as a "task", and until there is more experience is gained working with required activities or converting activities to tasks, it may be premature to update the activity stream standard with additional capabilities around "required" activities.

7. Architecture Overview Diagram

Activity Streams delivering content for required action based on an automatically triggered event.