Pega® 7

We could also call this "contingency-based computing".

Proactive Computing

Invention Disclosure for the Pega Patent Process

Ora Lassila & Jeff Rogers

2015-06-26

draft version #1

THE POWER TO SIMPLIFY™

Pega

Contents

Proactive Computing: General Idea (1)

- **Executive summary:** *We compute ahead of time while waiting for the user.* Computing can be done ahead of time before we know what the user will commit to. "Draft objects" can be created, only to be used if the user commits his/her prior choices; the objects can be discarded if the user decides to do something else.
- **What is the problem this solves:** Typically we perform time-consuming actions after the user clicks on a "save" or "submit" button. This creates the perception of a slow-executing system. Instead, much of the computation can be performed while the user is still pondering his/her choices, and once the user commits there is less work to be done.

Proactive Computing: General Idea (2)

- In case of missing information (e.g., the user has not yet fully filled in a form), multiple “draft objects” are created. This is (metaphorically) similar to certain interpretations of *quantum mechanics*: multiple futures are possible, and once choices are made the “un-chosen” paths disappear. In the context of our invention, these futures can be thought of as contingency plans.
- In BPM, multiple paths through a process flow can be (partially) computed in advance. Once the user makes a choice, one branch is chosen.

Some Prior Art

- Game-playing software (such as chess programs) do most of their exploration of future moves while the user (opponent) is considering his/ her next move.
- Modern (Web-based) document editing tools (e.g., Google Docs, Evernote, Microsoft OneNote) do not have a save button. Computing and communication involved in saving a document is done while the user is doing other things.

Simple Example: Computing while waiting for the user

- Currently, much computation/communication happens in Designer Studio once the user pushes the "save" button (e.g., validation of input data).
- Instead of doing all this work after the user submits the form, draft data can be created on the server while the user is still contemplating pushing the "save" button. Effectively, we are computing while the user is doing other things, having less to do once the user commits.
- Actions involving major side-effects that are hard to undo cleanly (e.g., committing DB transactions) are done once the user commits. But conceivably we could have an open, yet-to-be-committed DB transaction underway while the user is still thinking of his/her decision.

Complex Example: Proactively Computing Contingency Plans

- If a screen flow in a PRPC application has branches, draft objects can be computed for each branch proactively, with one chosen and others discarded once the user makes up his/her mind.
- If a form is missing data, contingency draft objects can be created for each combination of possible inputs (this makes most sense when the input choices are enumerable and finite). Conceivably, there can be multiple levels of branching in the contingency tree.
- The order of computation of contingency plans can be prioritized based on the likelihood of individual choices (e.g., using the probabilities already encoded in PRPC flows).