Design Document - Competence-based Adaptation Asset

TUGraz – T3.4C – Apache 2.0 – Client side

About this asset

This asset will be implemented as client-side component. It decides for a given domain model and a given competence state on how the game should proceed (expressed as next game situations, represented by identification strings). The asset architecture will prevent multiple asset creation; only one asset per game is needed.

Asset mechanics

A structure of game situations, consistent of a stating game situation and possible successors, are used to identify the next game situation most suitable for the player. A mapping between game situations and competences (located in the domain model) is used to calculate the distance between the current competence state of the player (gained from the competence assessment asset) and the game situations. This could for example be: Number of competences in the game situation not possessed by the player, if there are some and Infinity otherwise. A game situation with the shortest distance is recommended to be presented next.

Asset interfaces

- The current and next game situation identification string can be requested from the asset. A C# representation in the asset interface could be:

  ```
  string getNextGameSituationId()

  string getCurrentGameSituationId()
  ```

- The player performance within the game situation can be submitted as success/failure:

  ```
  void setGameSituationUpdate(Boolean success)
  ```

Asset dependencies/requirements

- The Domain Model Asset supplies the game situations used for the decision of which one of them should be next for a player.
- The Competence Assessment Asset is used to retrieve the current competence state of a player. Milestone s

Milestone 1

- t1.1: Creating the first version of the design document, defining the API and creating a dummy asset with the API implemented

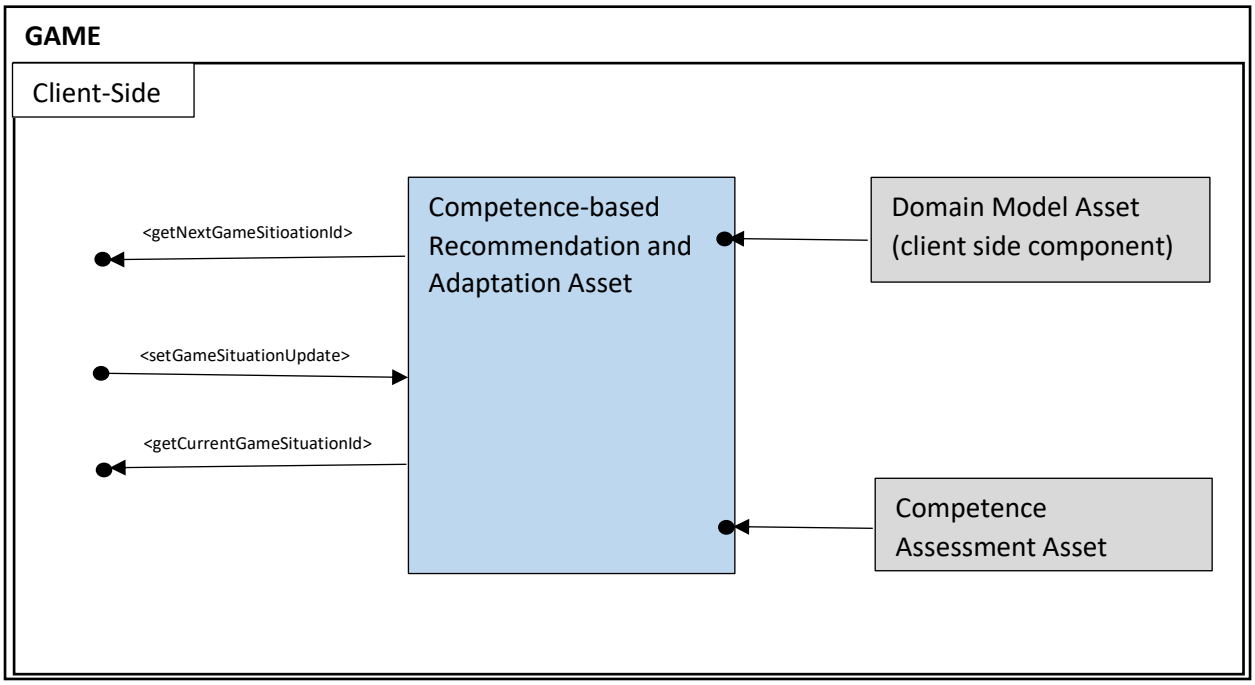
Milestone 2

- t2.1: Create Software Asset in line with Asset-Manager infrastructure.
- t2.2: Elaborate Settings-structure within the Asset-Manager infrastructure.
- t2.3: Include tracking functionality.

Milestone 3

- t3.1: testing the asset with a game
- t3.2: instructions and scripts for building and deploying

Graphical representation



Set up the Asset

For the cognitive intervention Asset, there are two thing to do (additionally to creating the Asset) when setting it up:

- The domain model asset and the competence assessment asset need to be in place, supplying a domain model and a competence state.

Use the Asset

The Asset recommends the next meaningful game situation in terms of learning.

- It is possible to request the next meaningful game situation, represented as an id string:

```
CompetenceBasedAdaptationAsset caa = CompetenceBasedAdaptationAsset.Instance;
String nextGameSituationID = caa.getNextGameSituationId();
```

- The current game situation can be requested:

```
String curGameSituationID = caa.getCurrentGameSituationId();
```

- A game situation can be completed successfully or not. This information can be returned to the asset via the following method:

```
caa.setGameSituationUpdate(true);
```

Deployment

For the source code the following GitHub-link can be used https://github.com/RAGE-TUGraz/CompetenceBasedAssets - it contains the Visual Studio solution of the competence based asset. Furthermore, the broken links to external asset DLLs need to be fixed for each project and the Bridge code need to be adopted to the new environment, e.g. changing the IDataStorage path.

For integration into Unity, the resulting DLLs need to put into a folder in the Unity working-directory.

Unit test

For executing unit tests, the source code need to be open in visual studio and all links need to be fixed. In the test-explorer all tests can be executed.