Assessment Strategy to ‘Future Proof’ Students as Computing Practitioners

Helen Partou (h.partou3@herts.ac.uk) & Lindsay Smith (l.l.smith@herts.ac.uk)
University of Hertfordshire, UK

Development > Rollout > Evolution

- **Motivation**
  - Team-based software development is core module delivery in computing at Hertfordshire
  - Students need relevant software engineering experience(s)
  - Previous software development platform not ‘fit for purpose’ teaching resource
  - Not compatible/upgradeable/adaptable
  - Overly complicated for ‘Zero to Hero’ student assessment in a six week development cycle

<table>
<thead>
<tr>
<th>Year</th>
<th>Research technical options &amp; feasibility of platform</th>
<th>Build platform &amp; supporting resources</th>
<th>Pilot platform as teaching tool with small L7 cohort</th>
<th>Larger scale rollout for L5 cohort(s) on-campus &amp; distance learning</th>
<th>Multiple module adoption &amp; evolution of delivery</th>
</tr>
</thead>
<tbody>
<tr>
<td>2014-15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2015-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2015-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2016-todate</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Assessment Strategies to Scope Student-based Solutions

- **Teaching resources are customised to support assessment**
  - Demonstration videos, FAQs and supervision supports instructional scaffolding as students gradually increase technical expertise.
  - An example ‘Orders’ application provides opportunities for formative feedback and minimises the student-tutor ‘expectation gap’ [2] of assessment deliverables.
- **Applications built in the platform are potentially scalable to any real-world scenario**
  - Supports constructivism, e.g. cinema film showings
  - Limitations for summative assessment include:
    - Managing trade-offs between case study complexity and platform functionality to define project scope
    - For example: matching deliverable technical competences with available timeframe
  - Summative assessment strategy has categorised marking criteria
    - **Baseline** = minimum engagement for a pass mark
    - **Advanced** = independent tasks gain higher marks
  - Example documentation for software: User Acceptance Tests (UATs)
  - Staff simulate client role to check software is ‘fit for purpose’
  - UATs support delegation of tasks to team members
  - Promoting “T-Shaped” individuals (specialised generalists) [3]

Purpose-built Platform as a Teaching Tool

- **Web-based**
  - We built an open-source development stack with an example ‘Orders’ system, utilising the Model-View-Controller (MVC) architecture for students to undertake data-driven web programming.

- **Portable & robust**
  - The tool is ‘plug-and-play’ and can be integrated with cloud-based tools.
  - Lightweight, compatible with multiple environments, re-usable and 100% reliable to-date.
  - Students can experiment with impunity.

Are Soft Skills Harder than Hard Skills in Software Development Projects?

- **Problems**
  - A STEM educational challenge is inherent complexity in delivering software development skills
    - Preparing students for employment in the computing industry
    - Employers cannot put graduates ‘in front of a client’ [1]
  - Teaching ‘hard’ skills focuses on technological constraints
    - Keeping up with technological change and advances
  - Teaching ‘soft’ skills focuses on team work
    - Student participation: passengers (lack of interest, engagement and/or feeling of inferiority) vs. diligent isolation (poor delegation, perfectionism and/or presence of passengers)

- **Solutions**
  - Reduction in technical complexity, e.g. robustness of platform enables ‘Zero to Hero’ solutions
    - Agile approach, staff development and staff-student feedback
  - Optimising teaching staff engagement with student teams
    - Managing student team autonomy
    - Student and staff teams collaboration
    - Team clinics, tutorial triage

Industry 4.0 and Future Developments

- **Current developments**
  - Technical
    - Exploring integration of the platform with Git-based systems, e.g. Azure DevOps, which facilitates sophisticated version control in the cloud.
  - Compassion-focused pedagogy (CfP) [4]
    - Supporting student team dynamics and task management.

- **Future developments**
  - Feasibility of adapting this approach to fast-moving technological change.
    - How the approach and/or platform integrates with, or could transfer to, other fields and technologies
    - Such as Internet of Things (IoT) e.g. ‘smart’/cognitive technologies/digitalisation.

References