ENGR 301 *Project Management*

Lecture 9 — GitLab III

Scope and Time Management in GitLab

James Quilty

School of Engineering and Computer Science
Victoria University of Wellington

Introduction

Today's lecture discusses how GitLab's *Milestones Epics* and *Iterations* features can help manage scope and time. Under PMBOK:

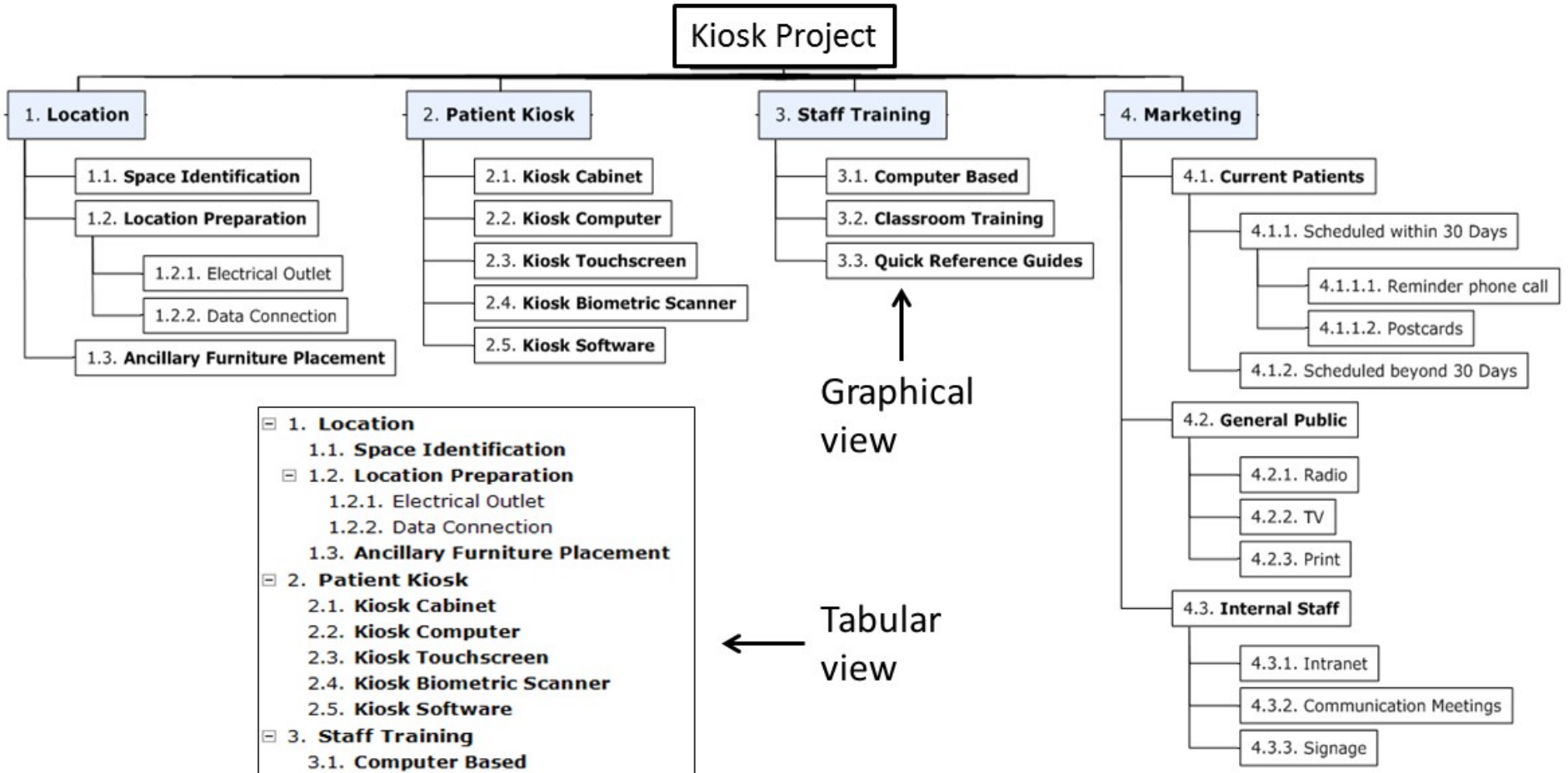
Scope Management Defines, and gains agreement and common understanding with stakeholders, on the work required to complete the project successfully.

Time Management Estimates how long it will take to complete the work, and develops an acceptable project schedule.

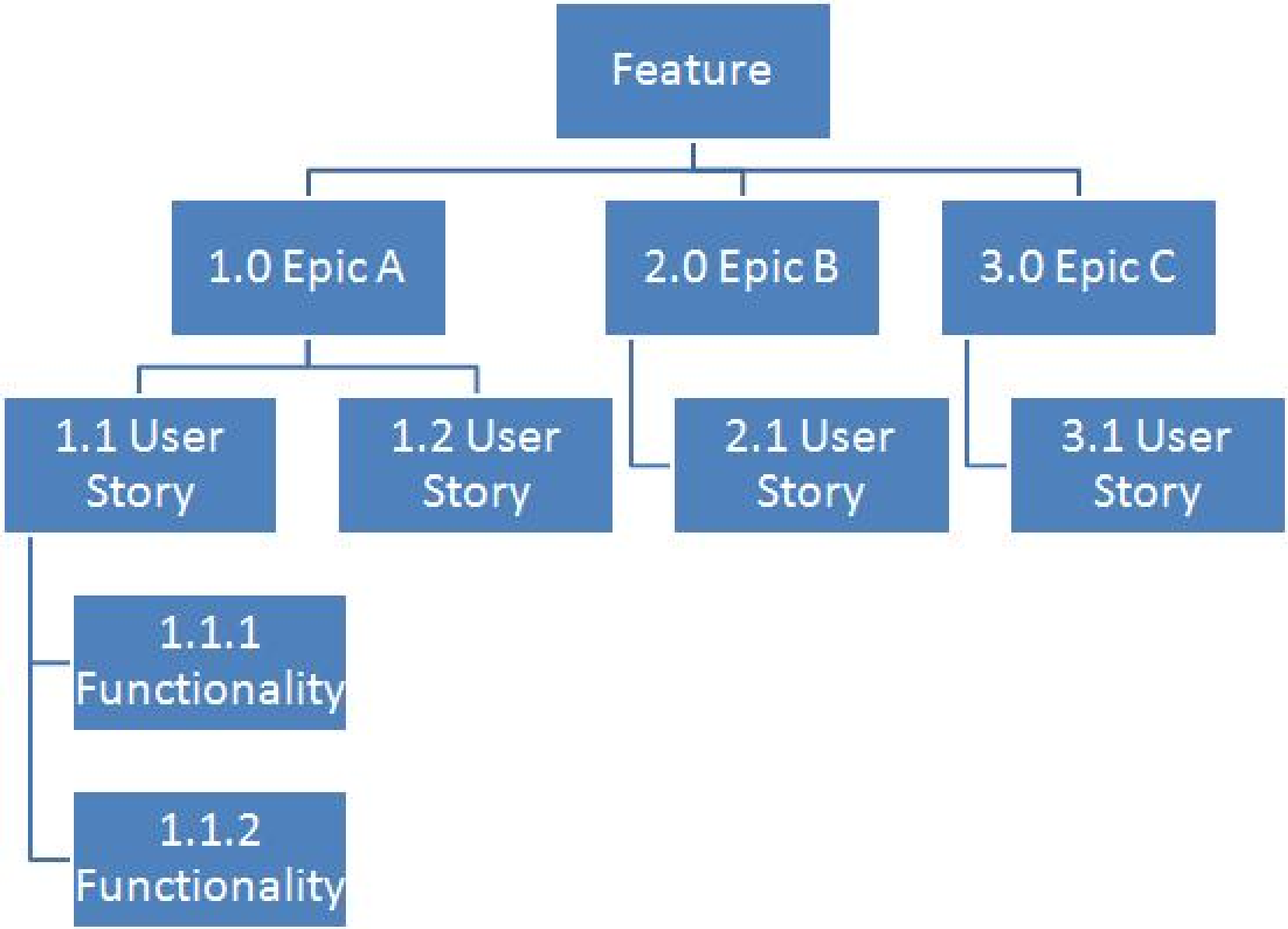
Please see *Introduction to Project Management*, 5th edition, by Kathy Schwalbe, Chapter 4 for additional detail.

Scope Management

Work Breakdown Structures (WBS) compose/decompose the project into Work Packages at the bottom-level. Note: there is no implied order or precedence!



Scrum WBS Analogue



Time Managment

PMBOK uses the concepts of *Activities* and *Milestones*.

Activities are distinct, scheduled portions of work, typically within a work package;

Milestones are significant *events* in a project;

Gantt Charts show a graphical representation of activities over the project.

Work packages and activities are intentionally separate in PMBOK, one concerns scope, the other time.

For a discussion of Gantt Charts, see: `https://bit.ly/4cer0Fc`

GitLab's Features

GitLab provides *Milestone Epic* and *Iteration* features which can be used to manage scope and time.

- *Iterations* composed of *Issues*
- *Milestones* composed of *Issues*
- *Epics* composed of *Issues*

The course methodology recommends *Milestones* be preferred over *Iterations*; *Epics* are a "special" group-level feature.

Scope Management

The main way of defining scope in GitLab is via the *Issue*. The question is, however, how to manage the overall scope of a project

- Top-down decomposition of project into *work packages* and *tasks*
- Bottom-up composition of *tasks* into *work packages* which make the project
- . . . ?

GitLab's Issues promote the second approach. What often occurs in practice is the third.

Time Management

GitLab provides several mechanisms for managing and tracking time, but poorly separates scope from time.

- Issues have due dates, time estimates and time expenditure
- Milestones have start and end dates
- Iterations have a *cadence*
- Epics have start and end dates

There's no clear way of composing a work package independent of time considerations, but several useful features for managing time.

Iterations (deprecated)

Iterations represent a periodic workflow, they have a *cadence*

- Iterations are a collection of Issues to be completed in the cadence period
- Issues roll-over from one Iteration to the next
- Issues can be related to only one Iteration at a time

Useful for BAU... not so useful for meeting project goals.

Milestones

The "milestone" is an idea from classical project management (PMBOK). GitLab’s Milestones are a bit different.

- Milestones are a collection of Issues to be completed within a fixed period
- A burn-down chart is displayed showing progress against time
- Issues can be related to only one Milestone at a time

These are actually pretty useful for managing project execution.

Epics

Epics are an idea from agile software development (Scrum) and are found at the Group level in GitLab

- Epics are a collection of Issues and other Epics to be completed within a fixed period
- Epics can be displayed in a Roadmap (Gantt Chart)
- Issues can be included in more than one Epic

Epics have some potential for constructing a *work breakdown structure* but their intrinsic limitations related to time and flat internal structure are frequently impediments.

Summary

Managing scope and time in GitLab is not as straight-forward as it could be.

- Constructing work packages as Milestones is good practice;
- Constructing Milestones corresponding to iterations (little “i”) is also good practice;
- Mirroring Epics from Milestones will help with a Roadmap view;
- Iterations are probably best avoided in favour of Milestones

Without time management via Milestones (and Epics), it will be hard to know whether or not project goals will be achieved.