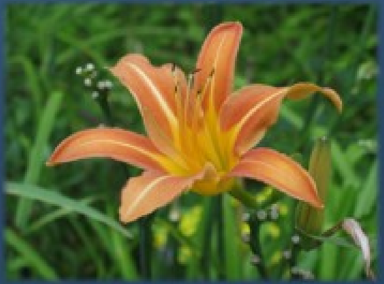
Efficient Python development with small teams

EuroPython 2013
Florence, Italy

Marc-André Lemburg

Marc-André Lemburg



Agenda

1. Introduction
2. Running Projects
3. Lessons Learned
4. Discussion

Agenda

1. Introduction
2. Running Projects
3. Lessons Learned
4. Discussion

Typical IT Project

- Specification
- Prototype in e.g. Python
- Design
- Implementation in e.g. C++/Java
- Deployment
- Support

Typical Python Project

- Idea/Specification
- Prototype in Python
- Design while you prototype
- Prototype turns into implementation
- Deployment
- Support

Advantages of Python Projects

- Easy to adapt to changing requirements
- Excellent time-to-market
- Smaller teams
- Overall lower costs

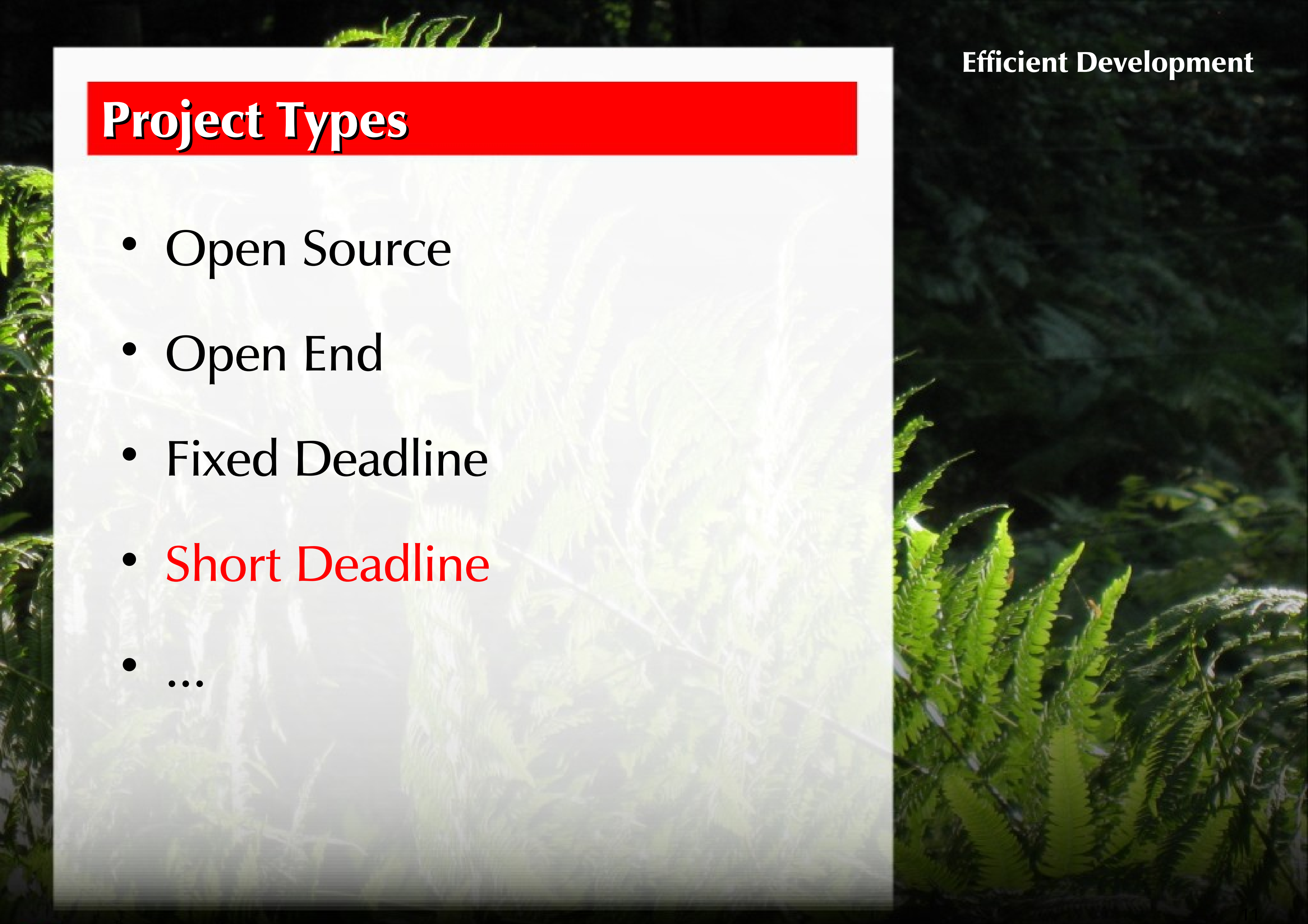
Problems with Python Projects

- No trial&error phase
- Design becomes important early
- Code has to be flexible
- Scalability has to be built in right from the start
- No linear project flow
- Good project coordination is key

Agenda

Running Projects

>>> Project Specs

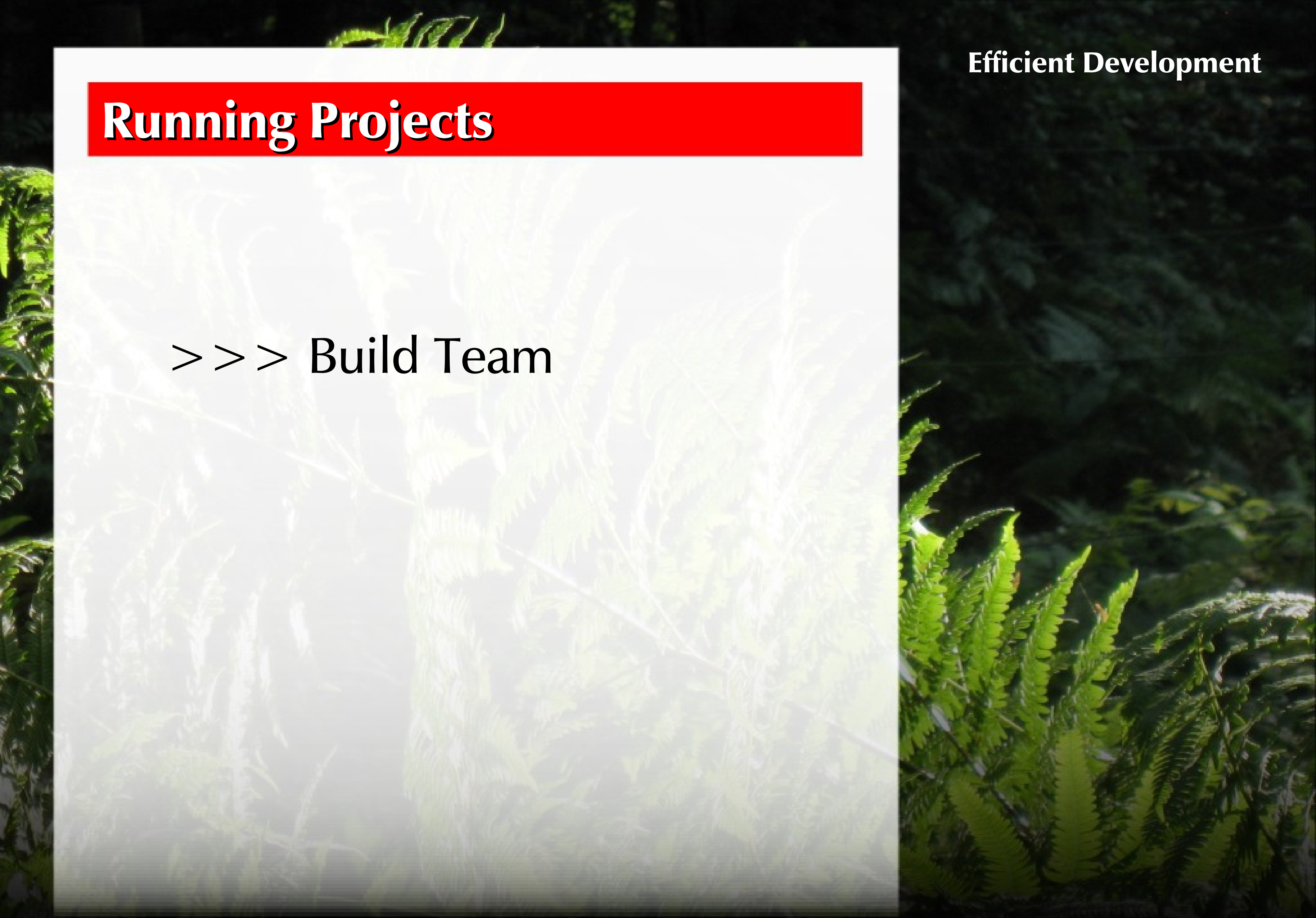


Project Types

- Open Source
- Open End
- Fixed Deadline
- Short Deadline
- ...

Project Specification

- Try to develop the specification as part of the project
- Adapt the specification as the project requirements change
- Educate the customer about needed specification decisions



Running Projects

>>> Build Team

Team Size

- 1 PM, 2-5 Developers
- 1 PM, 10 Developers (max)
- 1 PM, multiple teams

Team Location

- Office
- Remote
- Mix of both
- Synchronize office / remote members

Choosing Team Members

- PM who is at least as good as the developers
- Developers who can work on their own
- Developers who can manage themselves
- Efficient people

PM Team Motivation

- Set deadlines
- Be responsive
- Give feedback
- Work just as hard as the developers
- 'Thank you's help a lot

Running Projects

>>> Communication

Two Lines of Communication

- PM → Customer
- PM → Developers
- Avoid:
  Customer → Developers
  Developers → Customer

Customer Communication

- Regular status updates
- Project changes
- Developing / Updating project specification
- Manage deadlines
- Transparency

Project Communication

- Daily meetings
- Weekly meetings
- Meetings on demand, but constantly open chat window

Running Projects

>>> Initial Setup

Milestone Management

- Typical milestones:
  setup, concept, alpha, beta, release candidate, release
- Update/support releases
- One milestone every 2-4 weeks

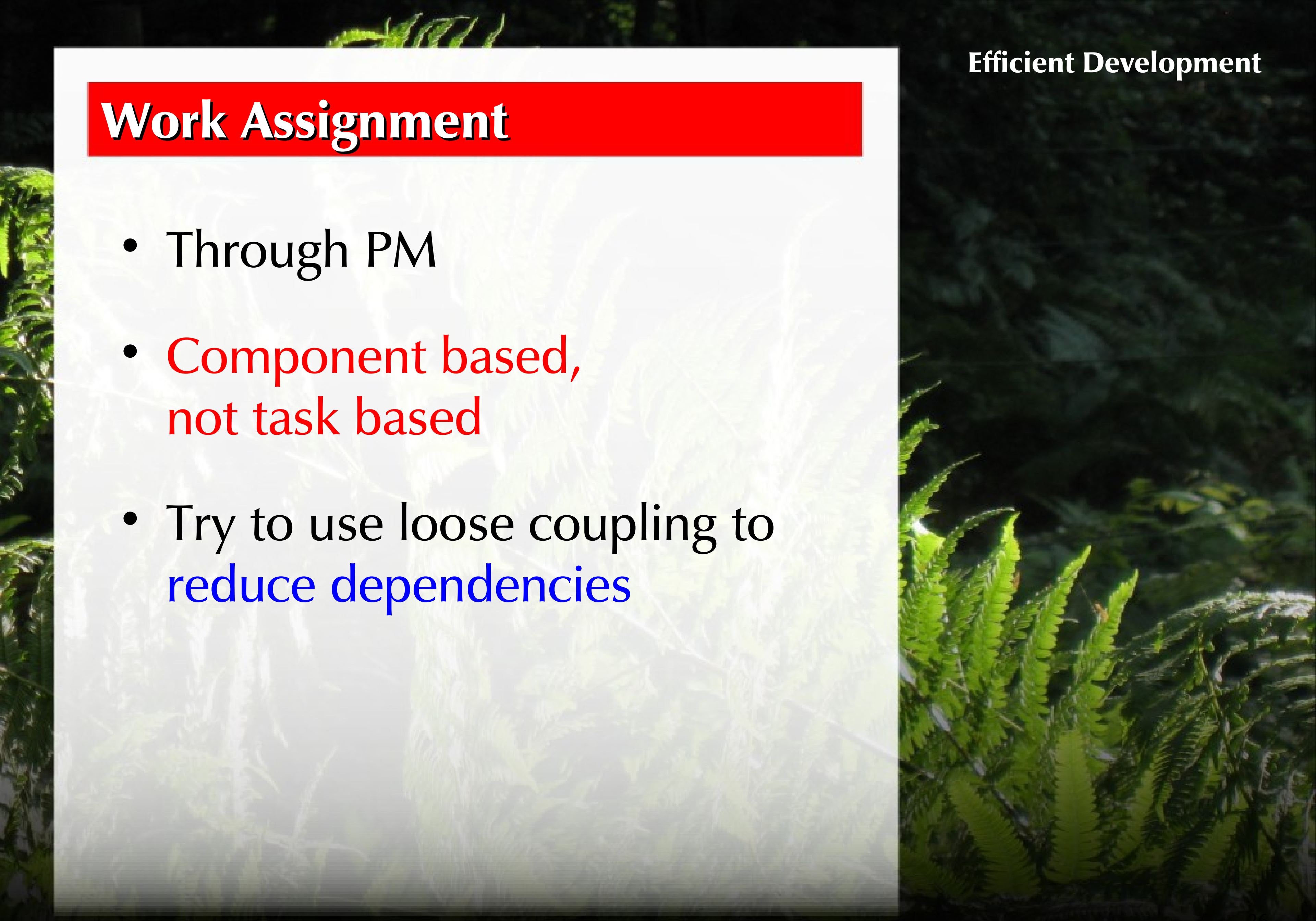
Project Documentation

- Wiki
- Text/ReST files
- Word/Google Docs files
- Whiteboard / Etherpad

Task Management

- Using ticket system
- One ticket per task
- Meta-tickets for larger tasks
- Ticket categories (Type of ticket, components, milestones)

Work Assignment



- Through PM
- Component based,
  not task based
- Try to use loose coupling to
  reduce dependencies

Running Projects

- >>> Work
- >>> Release
- >>> Support
- >>> Finalize

Agenda

1. Introduction
2. Running Projects
3. Lessons Learned
4. Discussion

Type of customer

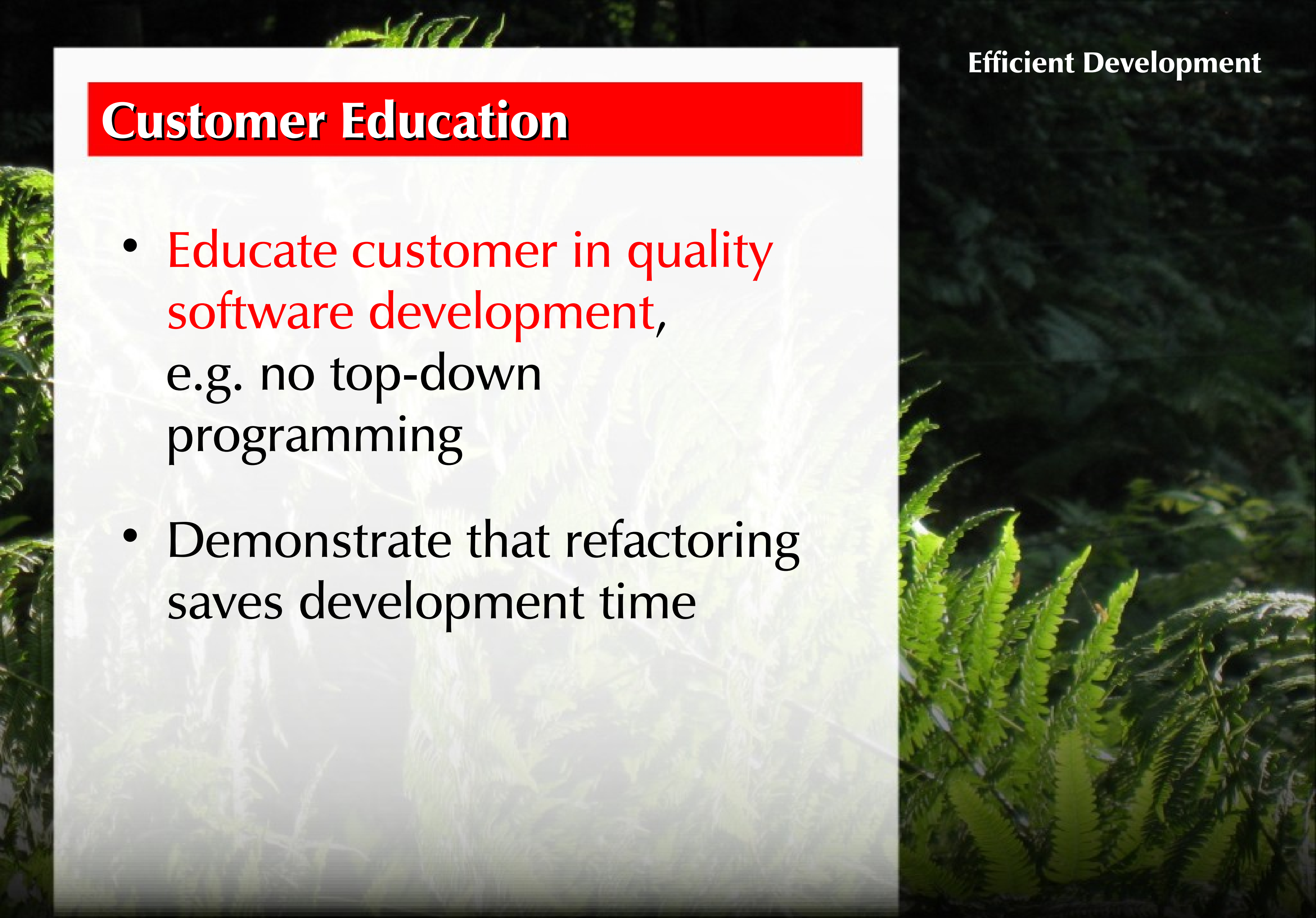
- Experienced, high expectations
- Great ideas,
  but no IT experience
- Sells solutions that don't yet exist
- ...

Customer Expectation Mgt.

- Clear communication
- Use safe time estimates,
  meet deadlines
- Define limits
- Help with defining milestones;
  find balance between expectation
  and development time

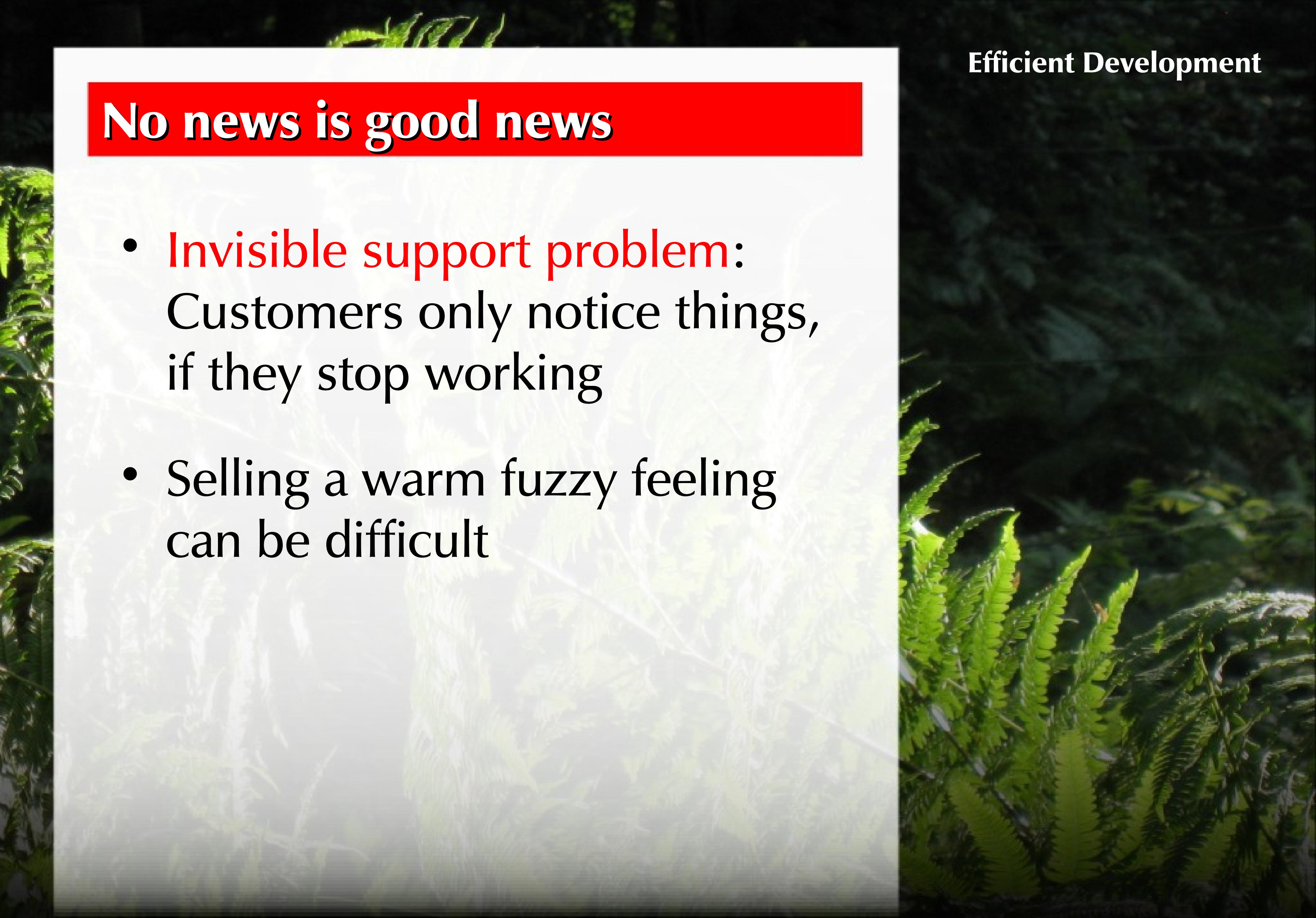
Customer Education

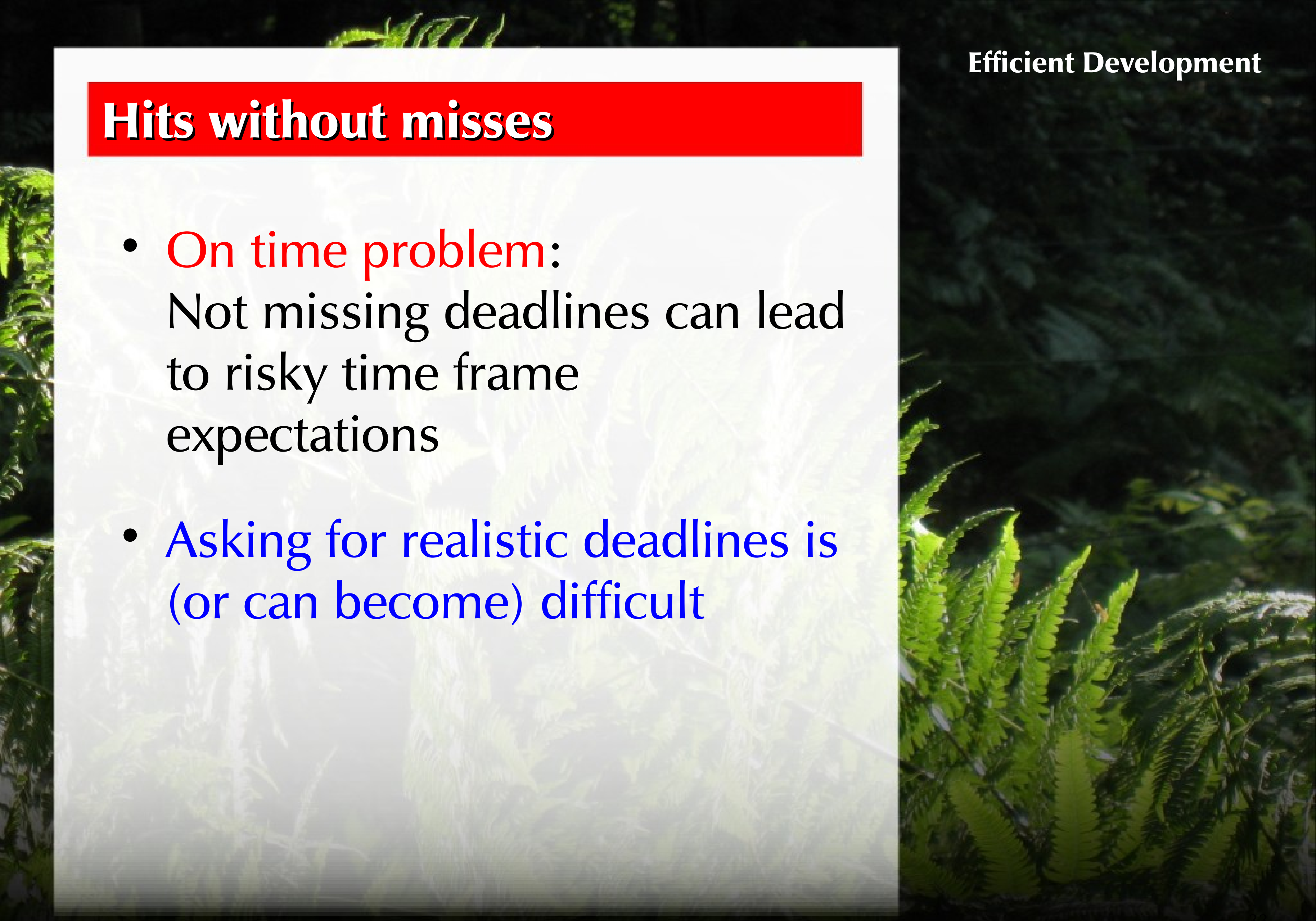
- Educate customer in quality software development, e.g. no top-down programming
- Demonstrate that refactoring saves development time



No news is good news

- Invisible support problem: Customers only notice things, if they stop working
- Selling a warm fuzzy feeling can be difficult





Hits without misses

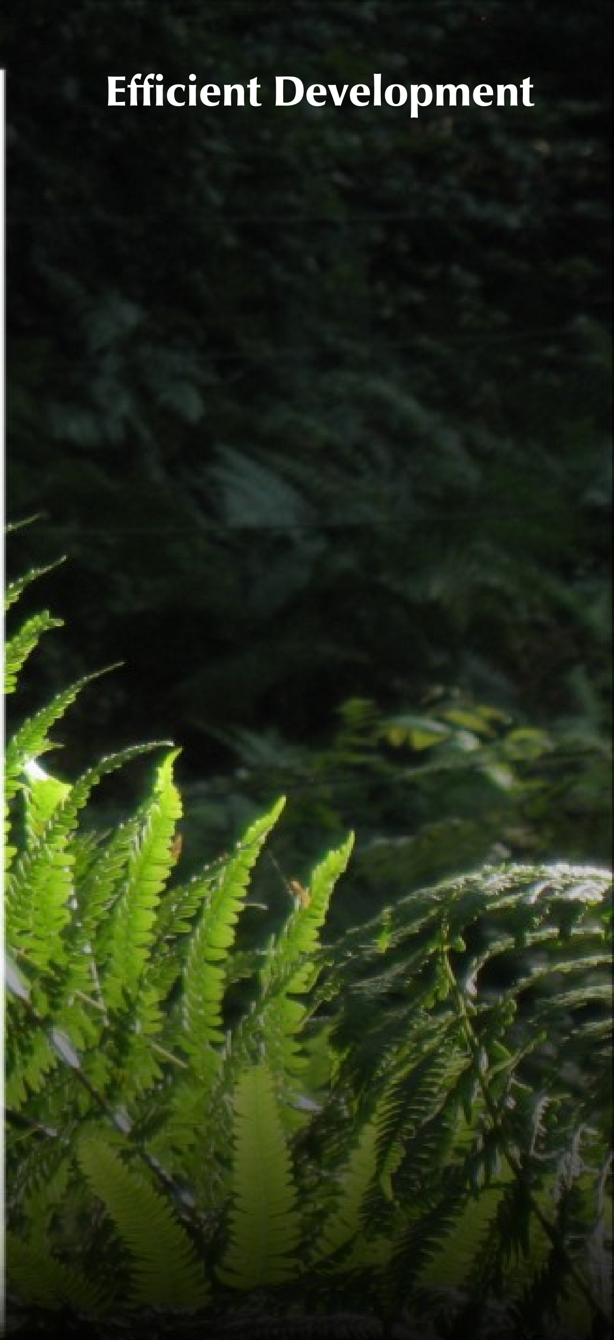
- On time problem:
  Not missing deadlines can lead to risky time frame expectations
- Asking for realistic deadlines is (or can become) difficult

Estimates

- Estimating project time is hard
- Lots of experience helps
- Use safety margins
- Consider possibility to scale up using more developers (higher costs for customer)

Estimates: A recipe

- Factor in demos, mockups, meetings
- Factor in refactoring
- Factor in unexpected complications
- Factor in the human factor
  (e.g. people getting sick)
- Factor in overhead due to changes
  in requirements
- Formula: $T_{estimate} = T_{expected} * (2 .. pi)$



Agenda

1. Introduction
2. Running Projects
3. Lessons Learned
4. Discussion

Questions ?

eGenix.com
Software, Skills and Services GmbH
Marc-André Lemburg
Pastor-Löh-Str. 48
D-40764 Langenfeld
Germany

eMail:	mal@egenix.com
Phone:	+49 211 9304112
Fax:	+49 211 3005250
Web:	http://www.egenix.com/

Thank you for your time.

For more information, please write to info@egenix.com or

visit our website at http://www.egenix.com/