

***FORGING A COMMUNITY – NOT:***

**EXPERIENCES ON ESTABLISHING AN  
OPEN SOURCE PROJECT**



---

Juha Järvensivu and Tommi Mikkonen  
Tampere U of Tech, Tampere, Finland



# Table of Contents

---

- Target environment
- Short history of project Laika
  - Becoming of age
  - Peak of fame
  - Decline
  - Epilogue
- Lessons learned
- Future research ideas
- Conclusions

# Target environment: Maemo



**Matchbox**

**GConf**

**Gnome VFS**

**GTK**

**Pango**

**GDK**

**X**

**Glib**

**D-Bus**

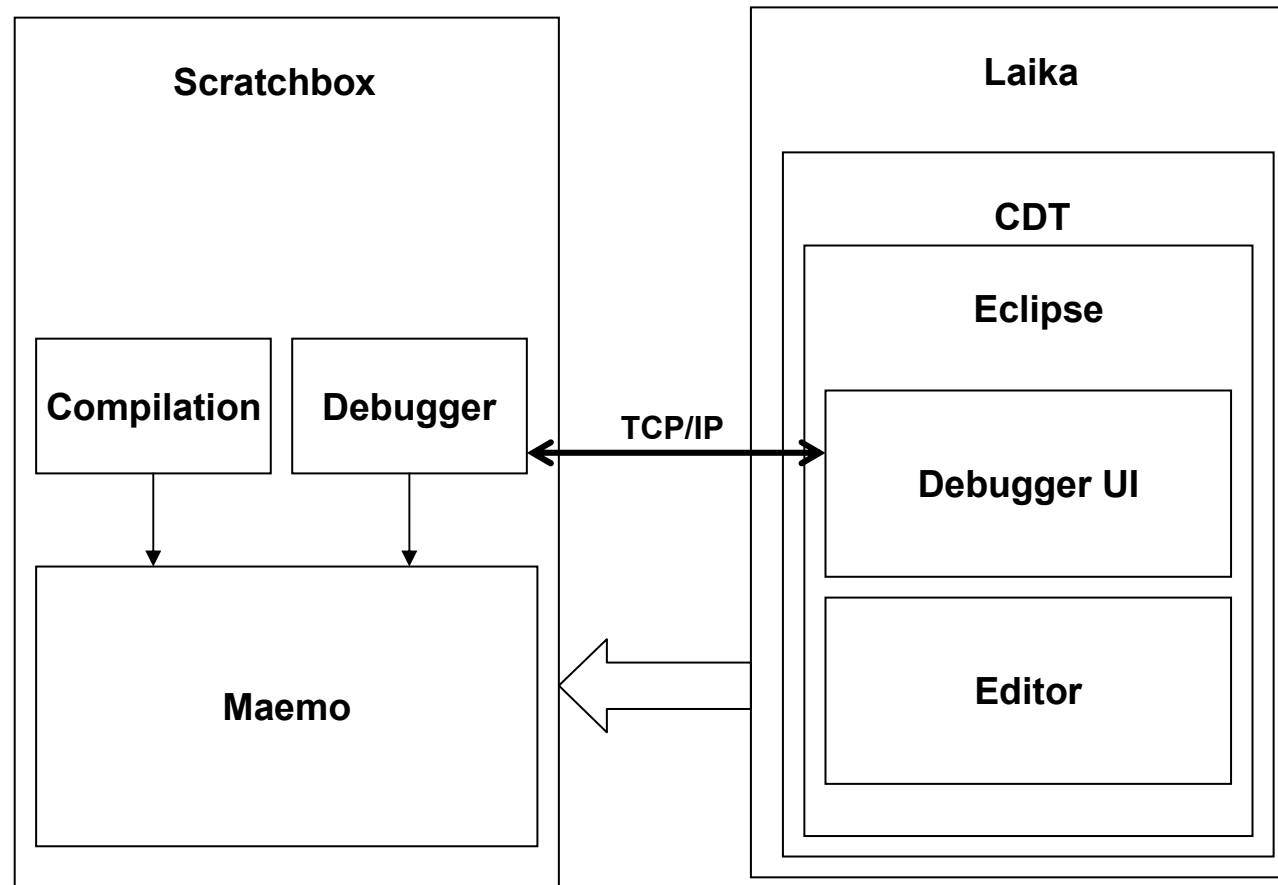


# Short history of project Laika

---

- Fall '04-Summer '05: Beginning
- Fall '05-Summer '06: Peak of Fame
- Fall '06-Summer '07: Decline
- Fall '07: Epilogue

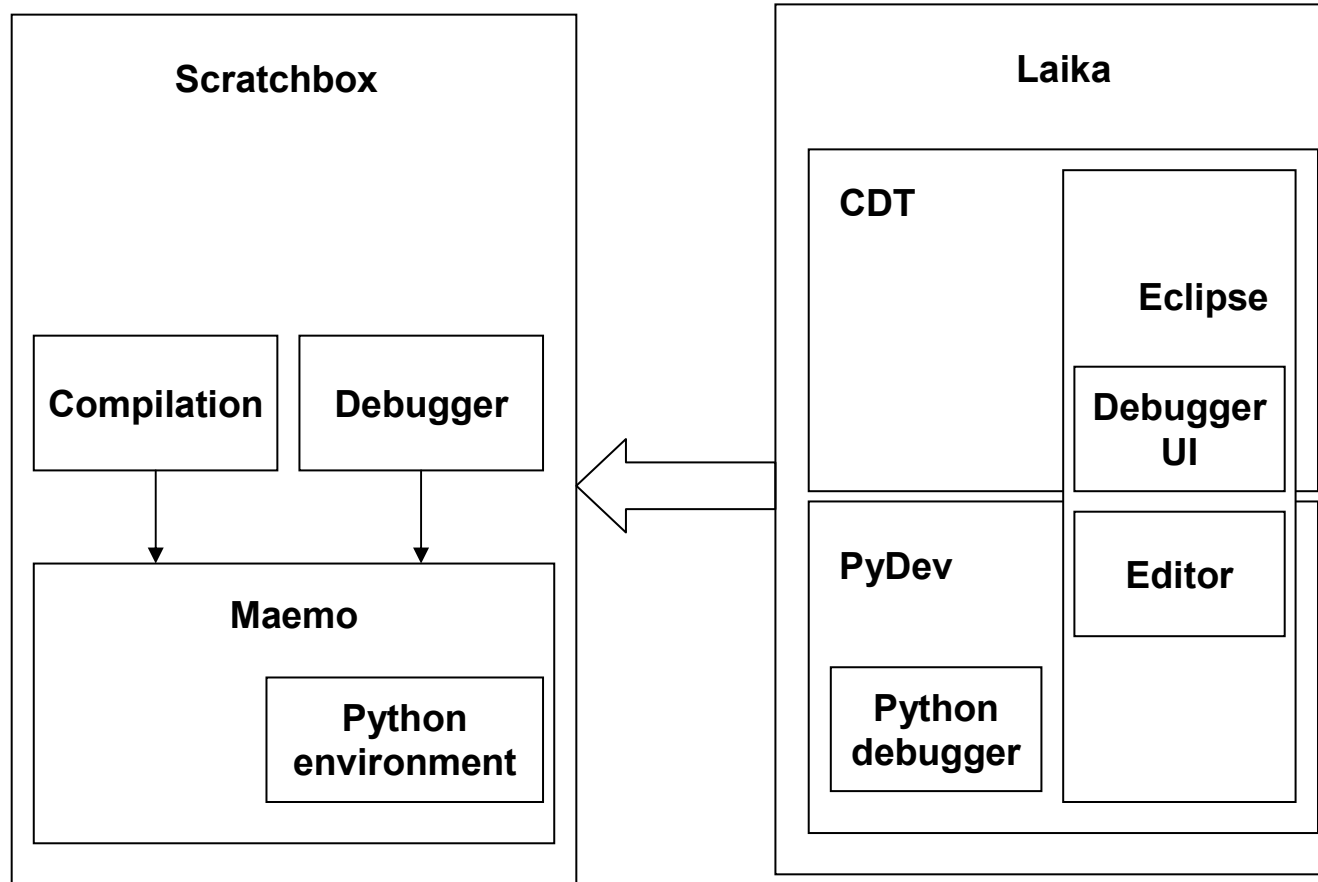
# Becoming of age



OSS'08, Milan, September 2008



# Peak of Fame





# Decline

---

- More and more effort on supporting users with different versions
  - Numerous components -> Even more numerous version combinations
- Less and less "own" feature development
  - What is the role of the community in the long run?



# Epilogue

---

- Merged with Esbox to form a new system authored by different developers



# Lessons learned

---

- Focused mission which remained undocumented
- Too much a debugging, integration, and testing project and too little a real development project.
- Developers interested in SDK development, not that much in Maemo programming
- Relying on agile principles that are not necessarily ideal for community building
- Dwarf in the league of giants



# Lessons learned

---

- Steep allocation of responsibilities but minimal staffing
- Subcontracting for money instead of community building for free
- Lack of active marketing, user support, and other related supporting facilities
- Single client community
- No experience on establishing an open source project



# Future research ideas

---

- What kinds of things are well-suited for a community of volunteers, what kinds of things require a company, and what needs real external paying clients?
- Is there a recipe for community creation to be followed?



# Conclusions

---

- Community-based development worked well when developing new features. But ...
  - *... it was hard to motivate developers volunteering in the project when tasks like integration, quality assurance, and testing was repeatedly requested.*
- We used agile practices that emphasize the importance of colocated teams. But ...
  - *... we should have embraced new developers by offering easy ways to practically participate in the development.*



# Conclusions

---

- Allowing funders and supporters to control a community seemed fruitful in the beginning. But ...
  - *... we entered a subcontracting mode, where funding was associated with features instead of supporting the community as a whole.*
- The number of readily available open source components that can be integrated in the system is so large. But ...
  - *... to keep the system manageable in the long run, it is essential to focus on a relatively small number of key features when the community is small.*



# Questions?

---

OSS'08, Milan, September 2008



# Euromicro 2009 Software Engineering and Advanced Applications (tentative)

---

- Special track for Open Source Software
- Athens, Sept. 2009



Thank you!

---