



Smart Objects For Intelligent Applications

SOFIA roadmap: Community Oriented Strategy

Legal disclaimer

The material contained herein is Confidential Information which may only be used in accordance with the terms of the SOFIA Project Consortium Agreement (PCA).

Access Rights needed for the execution of the SOFIA project are granted unless excluded in Annex IV of the PCA. No other licenses to any related IPR are implied.

The material is protected by copyright laws, and may not be reproduced, distributed or otherwise exploited in any manner without the prior permission of the rights holders.

The contributors are not liable for the use of this material except as stated in the SOFIA PCA.

Change History

Version	Date	Description	Affected Sections
0.1	2010-05-13	Initial version	All
0.2	2010-06-15	More content added	All
0.3	2010-06-22	Revision and new paragraphs	All
1.0	2010-06-30	Final version	All

List of Contributors

Participating Entity	Contributing Individuals
Indra	Elisa Gayol Cuervo (contributor & editor)
ESI	Raul Otaolea (contributor)
ESI	Alejandra Ruiz (contributor)

Table of Contents

1.	Introduction	4
2.	Objectives and main considerations	4
3.	SOFIA Community Processes	5
3.1.	Community Governance Processes.....	5
3.1.1.	Establish a community hierarchy.....	5
3.1.2.	Type of members (rights and duties).....	6
3.1.3.	Roles processes	6
3.1.4.	Internal policies definition.....	6
3.1.5.	Decision making Process.....	8
3.1.5.1.	Lazy Consensus	8
3.1.5.2.	Voting	9
3.1.5.3.	Types of Approval	10
3.2.	Support processes – QA Team.....	10
3.2.1.	Configuration management.....	10
3.2.2.	Validation process	11
3.2.3.	Incidents (problem resolution) management	11
3.2.4.	Change request (enhancements).....	11
3.3.	Management Process.....	12
3.4.	Documentation processes	12
3.5.	Business processes.....	13
3.6.	Marketing processes	14
3.7.	Acquisition Process	14
3.8.	Development processes	15
3.9.	Measurements processes.....	15
4.	SOFIA Community development checklist.....	15
4.1.	Technical Infrastructure	15
4.2.	SOFIA Community Name	16
4.3.	SOFIA Community Mission Statement.....	16
4.4.	Clarify that SOFIA Community is Open Source.....	16
4.5.	SOFIA Community Features and Requirements List.....	16
4.6.	Development Status	17
4.7.	Downloads.....	17
4.8.	Version Control and Bug Tracker Access	17
4.9.	Developer Guidelines	17
4.10.	Documentation.....	17
4.11.	Maintaining a FAQ	18
4.12.	Availability of documentation.....	18
4.13.	Developer documentation	18
4.14.	Example Output and Screenshots.....	18
4.15.	SOFIA Community Tone.....	18
5.	Summary.....	19
6.	References.....	19
	Appendix 1	20

SOFIA roadmap: Community Oriented Strategy		4 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

1. Introduction

The objective of this document is to describe how the SOFIA Community will be oriented and developed.

As the project targets to mobilize developers from various contexts, by the end of the project it is planned that the common application development framework will be used by all application domains in the project, and proven to be scalable and flexible for new ones. Hence, SOFIA ADK can be used for creating applications for any domain and their relevant run-time environments. The ADK should be able to handle several different information sources i.e. complex ontologies.

In order to fulfill this objective a community governance model must be defined to guarantee that a common approach and strategy is followed. This community will provide all the system stakeholders with resources for an efficient communication between them, during all the different SOFIA project stages, including a post project stage. The definition of the application development stack lifecycle, including central coordination of platform evolution, requirements and design, issues regarding ontology and DSL lifecycle and an easy resolution of conflicts of interest (internally/externally) will be a main point on this document. Additionally, all necessary activities for promotion, standardization and best practices for dissemination will be analyzed.

The present document provides the basis to define a strategy to follow to achieve a successful SOFIA's Community.

2. Objectives and main considerations

The main objective of this deliverable is to define a community creation and maintenance roadmap of SOFIA application development along and after the SOFIA project. This community must provide all system stakeholders with the resources for a successful communication and interaction with the rest of the roles defined in this document. In the scope of this task the most important sub-task is the definition of: The application development lifecycle, including central coordination of platform evolution, requirements and design, issues regarding ontology and DSL lifecycle and an easy resolution of conflicts of interest (internally/externally).

For the creation of a successful SOFIA application development community, the identification of the roles to be considered plays a crucial aspect on the T6.4 "Beyond SOFIA: Community". These roles that come from the specific domain expert designer (who adds new ontologies, modifies them, etc. regarding a specific domain) to the final user application developer must be defined. Every role will have some requirements and demands, offered by the community to be generated, which is the second main issue to be specified along this subtask.

It is worth to mention that the open source community will be able to access to the results of this development through the Eclipse Foundation. Other communities will be targeted, possibly implying a distributed SOFIA community.

3. SOFIA Community Processes

The SOFIA Community will be based on several processes. The main process will be the *Community Governance Process*, which will control the high decision making procedure and the creation of the rest of the processes. Each process will have a clear life-cycle and decision making workflow. In order to gain flexible processes, they can decide some aspects of their modus operandi.

The community will have several core processes like *Commercial process* to manage the relationships with companies, *QA process*, to control the quality of the developments, the *Documentation process*, to manage how the documentation about SOFIA is written and provided, *Core Development Architecture process*, responsible for managing the evolution of dependant and related technologies like Smart/M3 and Sofia ontology, the *Development Environment process*, which will manage the evolution of the ADK. Some processes would be shared by different work groups, for example the ones related to each domain (Personal Smart Environment team, Smart Indoor Environment team and Smart City team).

The processes are based on standard Quality Models like CMMI and SPICE.

3.1. Community Governance Processes

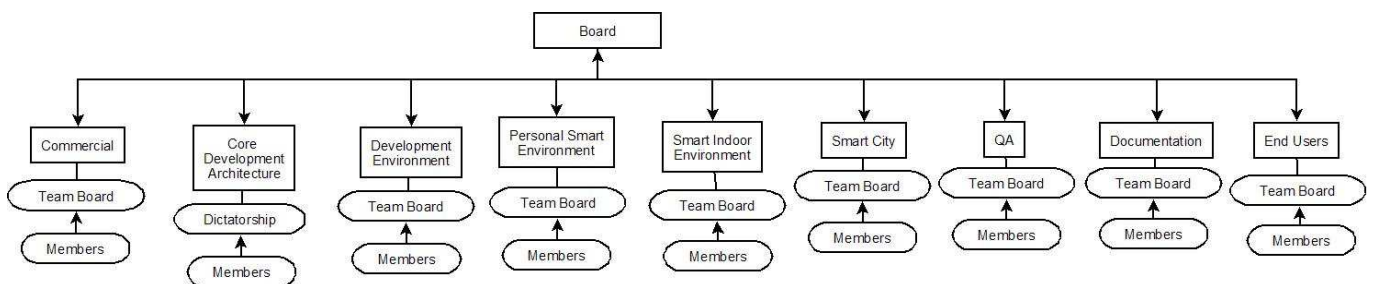
These processes are the ones related on how the community is managed. The main processes are:

- Establish community hierarchy.
- Type of members (rights and duties).
- Roles processes.
- Define internal policies.

These processes are explained in the following sections.

3.1.1. Establish a community hierarchy

Following it is shown a proposed hierarchy.



The community is divided into work teams each of them specialized on an area. The starting hierarchy can be see in the diagram above . The number of work teams and their organization could change in the future if the community strategy or members infrastructure decided so.

The whole community would be led by the community board. This board will have 3 members from the community chosen democratically. Every year, there would be an election period.

When a team is being created, the board will choose the team board for the first six months period. During this period, the team board should organize the team and prepare elections so the team members will choose their team board democratically. Only team members can be candidates for the

team board. One member could participate on more than one team.

3.1.2. Type of members (rights and duties)

Community members can be classified as following:

- Board members. Three community members can be board members.
- Team board members: In each team there are 3 people that have this role. (*) On Core Development architecture there will be only one person instead of three (dictatorship).
- Team members: Any voting member of a team.
- End Users: They use the community developments.
- Developers: They are occasional developers that haven't finished the "new member process" yet.

3.1.3. Roles processes

Board members: They should elect the team board when the team is created, should resolve the disputes among teams and have the final voice for the community strategy. They make decisions that affect the whole community. They are elected every year.

Team board members: They are elected by the team members. They decide the team strategy and the day to day work. They participate on the common strategy discussion. They can decide if the team is too big to be split or not. They accept new members for their team. They accept new members.

Team members: They have the right to vote for the team board and take part of day to day work discussion. They have fulfilled the *new member process*.

End Users: End users have voting right. They use the community developments. They inform about incidents and could ask for enhancements.

Developers: They are occasional developers that haven't yet finished the *new member process*. Their developments should be revised by prior members. They have no voting right.

3.1.4. Internal policies definition

Community Board elections

The community board is formed by 3 member of the community. These members are chosen every year after a democratic vote. All community members, no matter in which team work they are involved have the right to vote. Board Members can be reelected with no maximum times for reelections.

Three months after the one year period time expire, there will an open call for a month when any community member can put him/herself forward the candidates list for the board seats. After the month expires the list will be close and no more candidates will be accepted.

Each candidate is asked to publish personal information about the activities and task the candidate has done, his /her ideas for future goals to achieve, and testimonials for other members of the community that support the candidate.

SOFIA roadmap: Community Oriented Strategy		7 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

Once the candidates list is closed the voting time starts and will last one month. In this period every member of the community will receive a personal mail with the voting instructions and their personal key for the vote. The voting will be online and to enter the voting system, members are asked their key. On the voting system these are asked to arrange the list with their preference order. Once the voting time has expired the three more valuated candidates will form the elected board. During one month the current board and the new elected board will coexist in order to make the change smoothly.

Team Board/Dictatorship elections

Each work team will be led by a team board or by a dictatorship, depending on their own scheme. The election scheme is the same as the community board, but the voting members will be only the team members.

The complexion of the board is defined by the team board once it is agreed by the team members.

Three months after the one year period time expire, there will an open call for a month when any team member can put him/herself forward the candidates list for the board seats. After the month expires the list will be close and no more candidates will be accepted.

Each candidate is asked to publish personal information about the activities and task the candidate has done, his/her ideas for future goals to achieve and testimonials for other members of the community that support the candidate. Members from other teams have no voting right on board team or dictator election from a different team that their own, although they can be asked to give testimonial of a candidate from other team.

Once the candidates list is closed the voting time starts and will last one month. In this period every member of the team will receive a personal mail with the voting instructions and their personal key for the vote. The voting will be online and to enter the voting system, members are asked their key. On the voting system this are asked to arrange the list with their preference order. Once the voting time has expired the more valuated candidates will form the elected board. During one month the current board and the new elected board will coexist in order to make the change smooth.

Delete membership

A member can be deleted from any work team. A member can be deleted when he/she asks to be deleted, then he/she is automatically deleted.

A member can also be asked to leave when he/she acts against the community/team policies. If this happen, then the team should follow the decision making process that is explained in detail later, and should get to a unanimous consensus.

Members are related to a work team so if someone is deleted from all community work team is considered to be expelled from the community.

Creation of a new team

A new team can be created as a request of a group of the community members. This could happen when a team is too big to be handle by the board or a specific area became too important by itself. When this happens the members involved on the area/team make a request to the community board to create a new team. The community board will discuss this request with all the board/dictators of the teams. The creation would be accepted after a regular decision process where the community board and all the team boards will vote. The discussion will also include the new team board structure and

members.

When the community board accepts the creation of a new team, they also proposed temporary board members. These members have six months to define the team, the team strategy and organize elections. During this time community members can ask to be included on the new team. During this first six months to become a team member the only requirements are being a community member and send an email to any of the team board temporary members asking to be team member.

The temporary team board should discuss and get to an agreement with the new team members the new board structure, the requirements to be a team member and the first critical task. These agreements should follow the regular decision process.

On the six month the board should make an open call for candidates for the first elections. Candidates have 15 days to publish personal information about the activities and task the candidate has done, his/her ideas for future goals to achieve and testimonials for other members of the community that support the candidate. The last 15 days the new team members will receive a personal mail with the voting instructions and their personal key for the vote. The voting will be online and to enter the voting system, members are asked their key. On the voting system these are asked to arrange the list with their preference order. Once the voting time has expired the more valued candidates will form the elected board.

New Member

This process would be done by the team board as part of their duties. Each team board decides how this process is done.

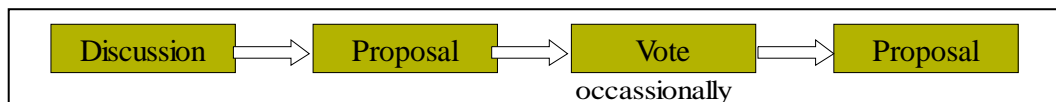
3.1.5. Decision making Process

This point describes different mechanisms for this decision making processes. Decisions about the future of the project are made through discussion with all members of the community, from the newest user to the most experienced member. All project management discussion takes place on the project/team mailing list (occasionally sensitive discussion occurs on a private list, but general project management discussion always occurs on the public lists).

These are the different activities that describe the process of the decision making.

3.1.5.1. Lazy Consensus

Decision making is usually a two step process. Occasionally a third step is required. These steps are:



Any community member can table an idea for consideration. In order to prompt a discussion about a new idea, an email is sent to the team/community list. Once the idea has become more concrete and appears to have a significant level of support within the community it is time to make a proposal. To make a proposal, a member summarizes the discussion in a new email with the tag "[Proposal]" at the start of the subject line.

Usually, gaining consensus within the community is easy since all community members have a common set of goals. By the time an idea reaches the proposal stage it is likely to have community support since it will already have been discussed public. However, full community consensus cannot be assumed at this stage, hence the need for a proposal stage.

SOFIA roadmap: Community Oriented Strategy		9 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

In general, as long as nobody explicitly opposes a proposal then it is recognized as having the support of the community. This is called lazy consensus, that is, those who have not stated their opinion explicitly have implicitly agreed to the proposal being implemented. The "[Proposal]" subject tag is used to grab the attention of community members who may have been unable to participate in the discussion phase. It is an indicator that a project decision is about to be made. This allows the community to assume that lazy consensus has been reached.

Lazy consensus is a very important concept within the project. This process allows a large group of people to efficiently reach consensus. This efficiency stems from the fact that someone with no objections to a proposal need not spend time stating their position; furthermore others need not spend time reading such mails.

For lazy consensus to be effective it is necessary to allow at least 72 hours before assuming that there are no objections to the proposal. This requirement ensures that everyone is given enough time to read, digest and respond to the proposal email.

The time spent gaining the consensus of the community does not prevent work proceeding on the idea. People are free to work on the implementation of any idea or proposal whilst the community examine and discuss it. This process need only be followed for an action that will significantly affect the project. Smaller actions, such as the addition of an optional feature or the fixing of a reported bug need no discussion. Members are free to action these items. Since this process is only followed for major activities the time spent reaching consensus is usually considerably less than the time spent implementing the decision.

3.1.5.2. Voting

Not all decisions can be made using lazy consensus. Issues such as those that affect the strategic direction of the project must gain explicit approval in the form of a vote. See the next section for a discussion of when a vote is needed.

If a formal vote on a proposal is called (signaled simply by sending an email with "[Vote]" in the subject line) then the following procedure applies.

All participants on the project/team list may express an opinion and vote by sending an email in reply to the original "[VOTE]" email with the following vote and information:

- +1 "yes", "agree" - also willing to help bring about the proposed action.
- +0 "yes", "agree" - not willing or able to help bring about the proposed action.
- -0 "no", "disagree" - but will not oppose the action going forward.
- -1 "veto", "disagree" - opposes the action going forward and must propose an alternate action to address the issue (or a justification for not addressing the issue).

To abstain from the vote then simply do not respond to the email.

Every member of the community can express their opinions in all discussion. If the discussion affects all the community then all community members can vote but if only affects a team then only the team members can vote. On a team discussion only team board have binding votes and on a community discussion only community board have binding voted for the purposes of decision making. The board members have the responsibility to ensure all opinions are considered.

When a [Vote] receives a "-1" (or veto), it is the responsibility of the community as a whole to address the objection. Such discussion will continue until the veto is either rescinded, overruled (in the case of a non-binding veto) or the proposal itself is altered in order to achieve consensus

(possibly by withdrawing it altogether).

In the very rare circumstance when team consensus cannot be achieved, the Community Board will decide the forward course of action.

3.1.5.3. Types of Approval

Different actions require different types of approval. The next table summarizes the types of approval. These range from a form of assumed consensus that allows work to progress unless it is challenged through to a majority decision of the project management committee. The following section describes which type of approval should be used in common situations.

Type	Description	Duration
Lazy consensus	An action with lazy consensus is implicitly allowed unless a binding -1 vote is received, at which time, depending on the type of action, a vote will be called. Note that even though a binding -1 is required to prevent the action all community members are encouraged to cast a -1 vote with supporting argument. Voting members are expected to evaluate the argument and, if necessary, support it with a binding -1.	72 hours
Lazy majority	A lazy majority vote requires more binding +1 vote than binding -1 vote.	72 hours
Consensus approval	Consensus approval requires 3 binding +1 votes and no binding -1 vote.	72 hours
Unanimous consensus	All of the binding votes that are cast are to be +1 and there can be no binding vetoes (-1).	120 hours
2/3 majority	Some strategic actions require a 2/3 majority of Community board, in addition, 2/3 of the binding votes cast must be +1. Such actions typically affect the foundation of the project (e.g. adopting a new code base to replace an existing product).	120 hours

3.2. Support processes – QA Team

These processes are managed by the Quality Assurance Team.

3.2.1. Configuration management

Based on CMMI configuration management-CM PA and SPICE SUP8 process, the following relevant points should be considered:

- Aspects to cover:
 - Identify components, version and releases with associated relevant items (like technical documentation).
 - Define clear base lines.
 - Control, modifications and releases.

SOFIA roadmap: Community Oriented Strategy		11 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

- Maintain configuration items history.
- Etc.
- Tools:
 - SVN, CVS, GIT...

3.2.2. Validation process

Purpose: To confirm that the requirements for a specific intended use of the component are fulfilled (based on SPICE SUP3 process and CMMI validation process area). Relevant points:

- Aspects to cover:
 - perform validation activities (testing the components directly or through application)
 - identify problems
 - provide validation data
 - make validation result available
 - etc.
- Tools:
 - SOFIA development environment tool → License Management Tool
 - Incidents tools (to report bug)
 - Selenium for web component
 - Junit, XUnit...

3.2.3. Incidents (problem resolution) management

Purpose: ensure that all discovered problems are identified, analyzed, managed and controlled to resolution (based on SPICE SUP9 process and part of CMMI-CM PA). Relevant points:

- Aspects to cover:
 - Define problem resolution strategy
 - Identify and record the problem
 - Analyse the causes of the problem
 - Evaluate the impact
 - Implement the problem resolution
 - Track status
 - etc.
- Tools:
 - Bugzilla, Mantis, ...

3.2.4. Change request (enhancements)

Purpose: Ensure that change requests are managed, tracked, and controlled (based on SPICE SUP10 and CMMI-REQM PA). Relevant points:

- Aspects to cover:

SOFIA roadmap: Community Oriented Strategy		12 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

- Define change request strategy
- Register the change request and status
- Assess the impact
- Approve changes
- Implement approved changes
- etc.
- Tools:
 - Same as for the incidents managements
 - OSMRT...

3.3. Management Process

This process is done by the Board and Team Board members.

Purpose: Identify, establish, coordinate and monitor components releases and versions deliveries.
Relevant points:

- Aspects to cover:
 - Publish periodically updated calendar.
 - Establish priorities among components.
 - Identify dependencies among component.
 - Etc.
- Tools:
 - Openproj, OTProject, etc.
 - Trac.

3.4. Documentation processes

These processes are done by the documentation team.

Purpose: To request and ensure maintenance of key technical documentation associated to each component. It is based on SPICE SUP7 and SUP4 and CMMI and SPICE engineering processes:

- Aspect to cover:
 - Identify technical documentation requested (* To be extended)
 - Establish documentation standards and requirements
 - Peer review of technical documentation (based on SUP4)
 - Etc
- Tools
 - Alfresco, DocMGR, Epiware, Wiki

SOFIA roadmap: Community Oriented Strategy		13 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

- (*) Possible Requested Technical documentation per component:
 - System/Software requirements document
 - Use cases
 - Architecture of components
 - Detailed design
 - Implemented code
 - Test plan
 - Test cases, etc

The technical documentation will cover all the development cycle from requirements documentation till test plan. The components owner should provide initially not only the code but also the requested technical documentation.

3.5. Business processes

These processes are done by the community board:

- Define a community financing schema.
- Define the processes related with the decided schema.
- (Proposal) For the first five years the board will be the one in charge of the expenses done and the sum of the year expenses will be divided among the companies related with the community. While SOFIA project is receiving funding, part of that funding will be inverted on the community.
- (Proposal) merchandising and donations.

In order to maintain SOFIA objectives beyond the end of the Artemis project, we propose to create a foundation to help the incipient community. SOFIA open source community have just come to life and in order to help the community, SOFIA Foundation would be created.

SOFIA Foundation will be based on Eclipse Foundation as it has a remarkable reputation on how a community with individuals, users, developers and organization are taken into consideration. SOFIA Foundation will be founded by dues from its members and governed by a Management Team. This team will be formed by strategic users, strategic software providers and organization from Personal Smart Environment, Smart Indoor Spaces, Smart City, User interaction and Smart environment Architecture dominions

The pillars of SOFIA Foundation would be: IT Infrastructure, Legal, development and Marketing & Promotion. SOFIA Foundation will have staff taking care of each of the pillars:

- **IT Infrastructure.** SOFIA Foundation will take care of the servers and the IT infrastructure needed to maintain SOFIA project and community. Here it is included the SVN code repositories, the development and support oriented mailing lists, project TRAC, download site, web site.
- **Legal.** The foundation will provide an established framework for intellectual property and financial contributions that simultaneously limits contributor's potential legal exposure. Just as Eclipse Foundation is focus on enabling the use of open source technology in commercial software products and services and consciously promote and encourage software vendors to use Eclipse technology for building their commercial software products and services so does SOFIA

Foundation with SOFIA technology.

In order to achieve this, the foundation will ensure that all contributions are made by the rightful copyright holders. Related to this, all code that is not created inside the community following the development process must complete the acquisition process where legal issues are particularly analysed.

- **Development.** SOFIA Foundation will take care of the developments done on the project and will participate in those areas that are considered strategic by the Foundation Management team. The foundation will provide services and support for the projects to help community achieve its goals of creating quality and reliable software.
- **Marketing & Promotion.** One of the SOFIA Foundation goals will be to promote SOFIA technology and the services and applications derived from it. SOFIA Foundation will organize activities, including co-operative marketing events with member companies, community conferences and other programs to promote the entire SOFIA community

3.6. Marketing processes

These processes are done by the commercial team:

- Define the community image.
- Define audience (who is interested on SOFIA).
- Define the communication channels and community presence (virtual and physical) on remarkable events in order to get to known SOFIA community.

Once the SOFIA Community is presentable it will be announced to the world by targeting:

- Partner's websites and intranets.
- Press Releases and articles.
- Other R&D projects.
- The Community will be submitted to different portals (e.g.: <http://freshmeat.net/>).
- SOFIA project publications (e.g.: SOFIA Newsletter 1st issue will include a section about the SOFIA Community).
- SOFIA Community url will appear on all project public documents (posters, presentations...).

All these activities will be developed in coordination with WP7 in order to boost SOFIA Community dissemination.

3.7. Acquisition Process

These processes are done by the Core Development Architecture, Development Environment, Personal Smart Environment, Smart Indoor Environment and Smart City Teams.

It is very much based on the community governance and is responsible for ensuring the quality and completeness of the component/development to be integrated in to the SOFIA platform:

- Aspects to cover:
 - Component Applicant.

SOFIA roadmap: Community Oriented Strategy		15 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

- Evaluation of applicants.
- Verification and validation of components.
- Legal issues.
- Acceptance of the component.

3.8. Development processes

These processes are done by the Core Development Architecture, Development Environment, Personal Smart Environment, Smart Indoor Environment and Smart City Teams.

It is also very much based on the community governance. In order to agree on the development process, we should study how this process is implemented on other communities. On the Appendix I we have include some studies done on Debian, Ubuntu, Mozilla and Linux kernel communities. After the study for the advantages and disadvantages SOFIA Community should agree on:

- Changes, new code, enhancement acceptance (development cycle).
- Releases development.

3.9. Measurements processes

These processes would be done by the board and team board members. The result of these processes will help to define not only the common strategy but also the team specific strategy.

Based on SPICE MAN.6 and CMMI-MA PA several points must be taken into account:

- Aspects to cover:
 - Identify measures needs based on community objectives.
 - Specify relevant measures: No. Errors, No. Enhancements, No. Downloads, No. Application using a component.
 - Collect, stored and analyze measurement data.
- Tools:
 - Pentaho, mysql.

4. SOFIA Community development checklist

This section contains main points' checklist to take into account before developing the SOFIA Community. The final strategy will be decided by the board members once it has been discussed with all the voting members (1).

4.1. Technical Infrastructure

SOFIA's community set of tools will include:

- **Web Site:** SOFIA Community web site will give a clear and welcoming overview of the project,

SOFIA roadmap: Community Oriented Strategy		16 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

and will gather the other tools (the version control, system, bug tracker, etc.). SOFIA Community website will have user login functionality but it will also permit read-only actions, some data entry actions, especially in the bug tracker and wiki pages to non logged-in visitors.

- **Mailing Lists:** SOFIA Community will use list management software with the following features: email- and web-based subscription, for subscribing and unsubscribing users when they request, moderation (to make sure they are a) not spam, and b) on topic), mailing list in digest versus message-by-message form, standard list and project information by means of auto-responders; administrative interface; header manipulation, archiving with prompt updating, referential stability, backups, thread support, searchability, etc.
- **Version Control:** SOFIA Community will use a version control system.
- **Bug Tracker:** The bug tracker will enable developers to keep track of what they're working on, coordinate with each other, and plan releases. The Bug Tracker will enable everyone to query the status of bugs and record information (e.g., reproduction recipes) about particular bugs. It is planned to be used for tracking not only bugs, but also tasks, releases, new features, etc.
- **IRC / Real-Time Chat System:** A place for quick, lightweight discussions and question/answer exchanges will be available at the SOFIA Community.
- **RSS Feeds:** SOFIA Community will offer at least one RSS feed on the front page, for sending out major announcements such as releases and security alerts to subscribers.
- **Wiki:** SOFIA Community will run a wiki, having a clear page organization and pleasing visual layout, so that visitors (i.e., potential editors) will instinctively know how to fit in their contributions.

4.2. *SOFIA Community Name*

SOFIA consortium will give an attractive and descriptive name to its community. A name that will give some idea what the community will do, easy to remember, different to some other projects/communities name, making sure that it does not infringe on any trademarks.

4.3. *SOFIA Community Mission Statement*

SOFIA consortium will use a quick description, a mission statement, so possible users can decide whether or not they're interested in learning more. This will be prominently placed on the front page, right under the project's name.

4.4. *Clarify that SOFIA Community is Open Source*

SOFIA Community will make clear, right below the mission statement, that the Community is "open source software" and give the exact license (the license to follow is still under discussion) More detail than the mission statement, (e.g.: some user or developer downloadable documentation) will be also provided within the front page.

4.5. *SOFIA Community Features and Requirements List*

SOFIA Community will specify a brief list of the features the software supports, this is, a logical expansion of the mission statement, and the kind of computing environment required to run the software.

4.6. Development Status

In order to give updated information, SOFIA Community is planning to include a development status page listing the project's near-term goals and needs and giving a history of past releases, with feature lists, so visitors can get an idea of how the community is progressing.

4.7. Downloads

SOFIA Community software will be downloadable as source code in standard formats. The distribution mechanism will be as convenient, standard, and low-overhead as possible. Software will conform to standard build and installation methods.

SOFIA Community will play special attention to releases version numbering, giving a unique number to each release.

4.8. Version Control and Bug Tracker Access

SOFIA Community in planning to provide real-time access to the latest sources using a version control system. Anonymously accessible version controlled sources will be also available when possible in order to make an effort to give people what they need to participate.

A well-maintained bug database visible from the front page is also planned. Bug tracker will be also used to track not only software bugs, but enhancement requests, documentation changes, pending tasks, and more.

4.9. Developer Guidelines

SOFIA Community will also define developer guidelines for explaining how the developers interact with each other and with the users, and ultimately how things get done. This initial document will be continuously updated within the community progress and it will adjust its language to reflect this evolution.

The basic elements of SOFIA Community developer guidelines will be:

- pointers to forums for interaction with other developers
- instructions on how to report bugs and submit patches
- some indication of how development is usually done

4.10. Documentation

SOFIA Community considers that documentation is essential and it will provide documentation for initial users: information about how to quickly set up the software, an overview of how it works, some guides to doing common tasks. This documentation will have an easy-to-edit format such as HTML, plain text, Texinfo, or some variant of XML.

Initial documentation scope will be defined covering the following minimal criteria:

- Telling the reader clearly how much technical expertise they're expected to have.
- Describing clearly and thoroughly how to set up the software, and somewhere near the beginning of the documentation, tell the user how to run some sort of diagnostic test or simple command to confirm that they've set things up correctly. Startup documentation is in some ways more important than actual usage documentation.

SOFIA roadmap: Community Oriented Strategy		18 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

- Producing at least one tutorial-style example of how to do a common task.
- Labeling the areas where the documentation is known to be incomplete.

As an accurate accounting of known deficiencies is the norm in the open source world, SOFIA Community will identify them (whether in the documentation, in the bug tracking database, or on a mailing list discussion).

4.11. Maintaining a FAQ

A well-maintained FAQ ("Frequently Asked Questions" document) is expected to grow within the Community in order to answer users and developers questions, evolving over time in response to people's day-to-day usage of the software.

4.12. Availability of documentation

Documentation is planned to be available from two places: online (directly from the web site), and in the downloadable distribution of the software.

4.13. Developer documentation

Developer guidelines will tell programmers how to get along with each other and developer documentation will tell them how to get along with the code itself.

4.14. Example Output and Screenshots

The SOFIA Community will also provide if needed screenshots or just files to ease user's participation and interaction.

SOFIA Community is also planning to include a news page, a project history page, a related links page, a site-search feature, etc.

4.15. SOFIA Community Tone

SOFIA Community will avoid private discussions (unless a specific need for privacy comes up) because:

- The discussion will help train and educate new developers.
- The discussion will train the Consortium in the art of explaining technical issues to people who are not as familiar with the software as it is.
- The discussion and its conclusions will be available in public archives forever after, enabling future discussions to avoid retracing the same steps.

SOFIA Community founders will maintain a zero-tolerance policy towards rude or insulting behavior in forums, never letting bad behavior slide by unnoticed and keeping forums friendly.

5. Summary

By the end of the project we will have a complete roadmap with the key topics to ensure an active developer community from the participating organization and affiliates. developing new services with the same technology and development tools and paradigm.

Important points like business model, type of governance, funding schemas will be clear and reached to a consensus. A flexible and scalable process approach will allow manage multi-domain markets covering critical issues like decision making workflows, responsibilities, and stake-holders interactions.

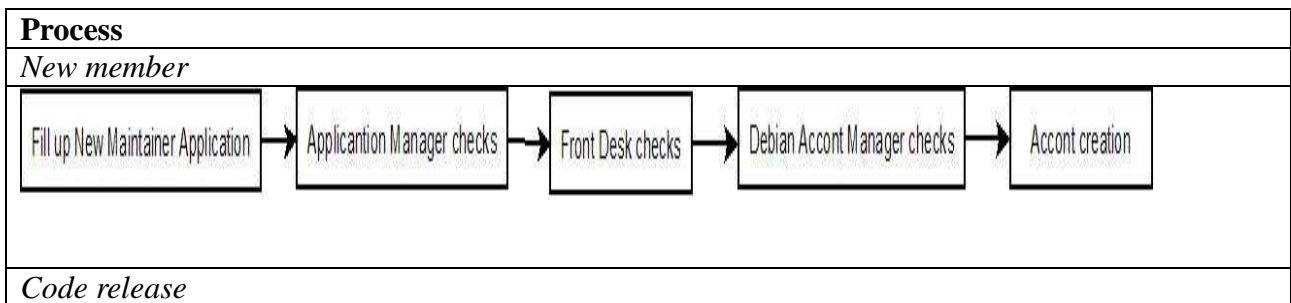
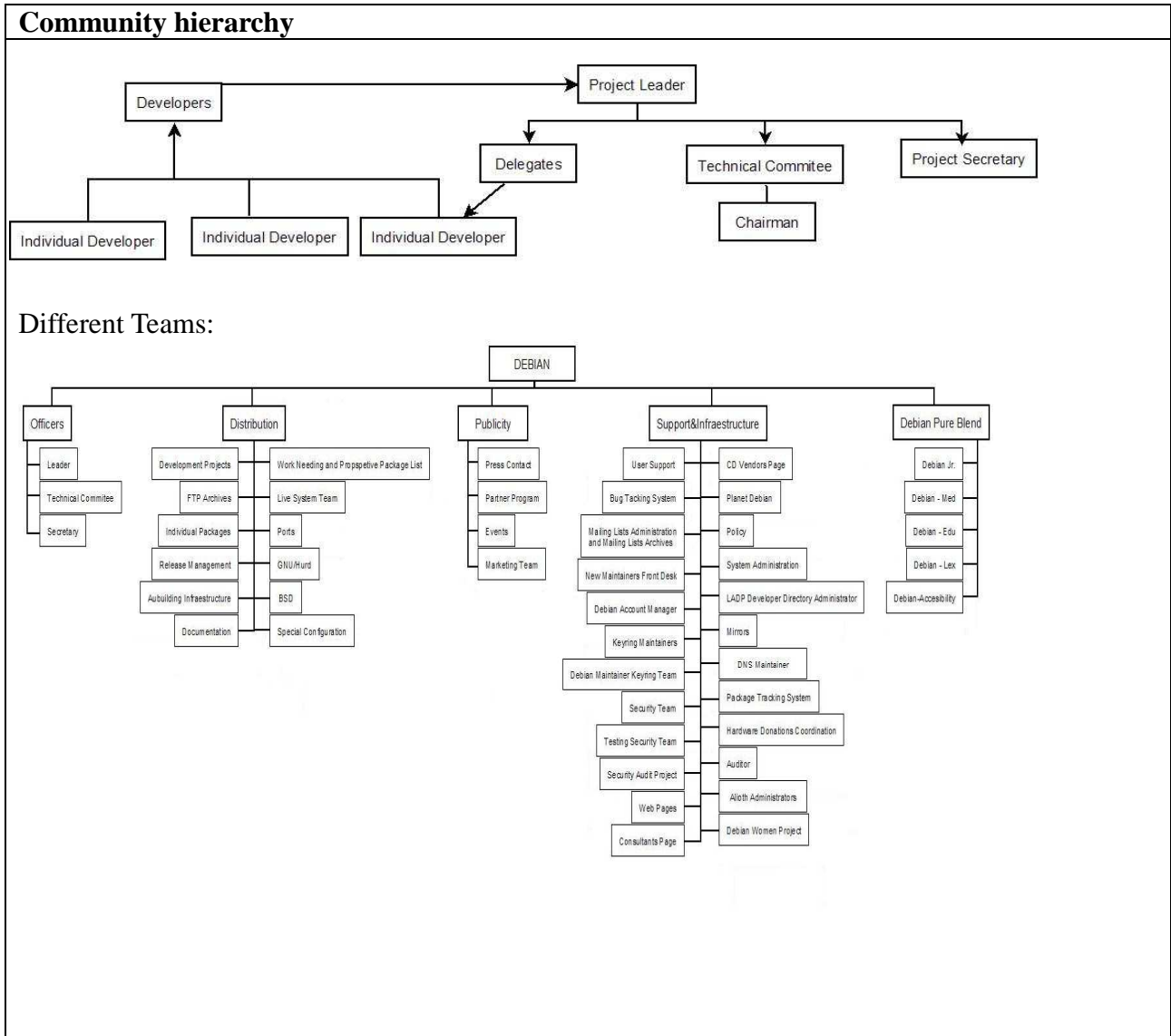
In summary, this community could be successfully materialised following the above mentioned processes and checked list.

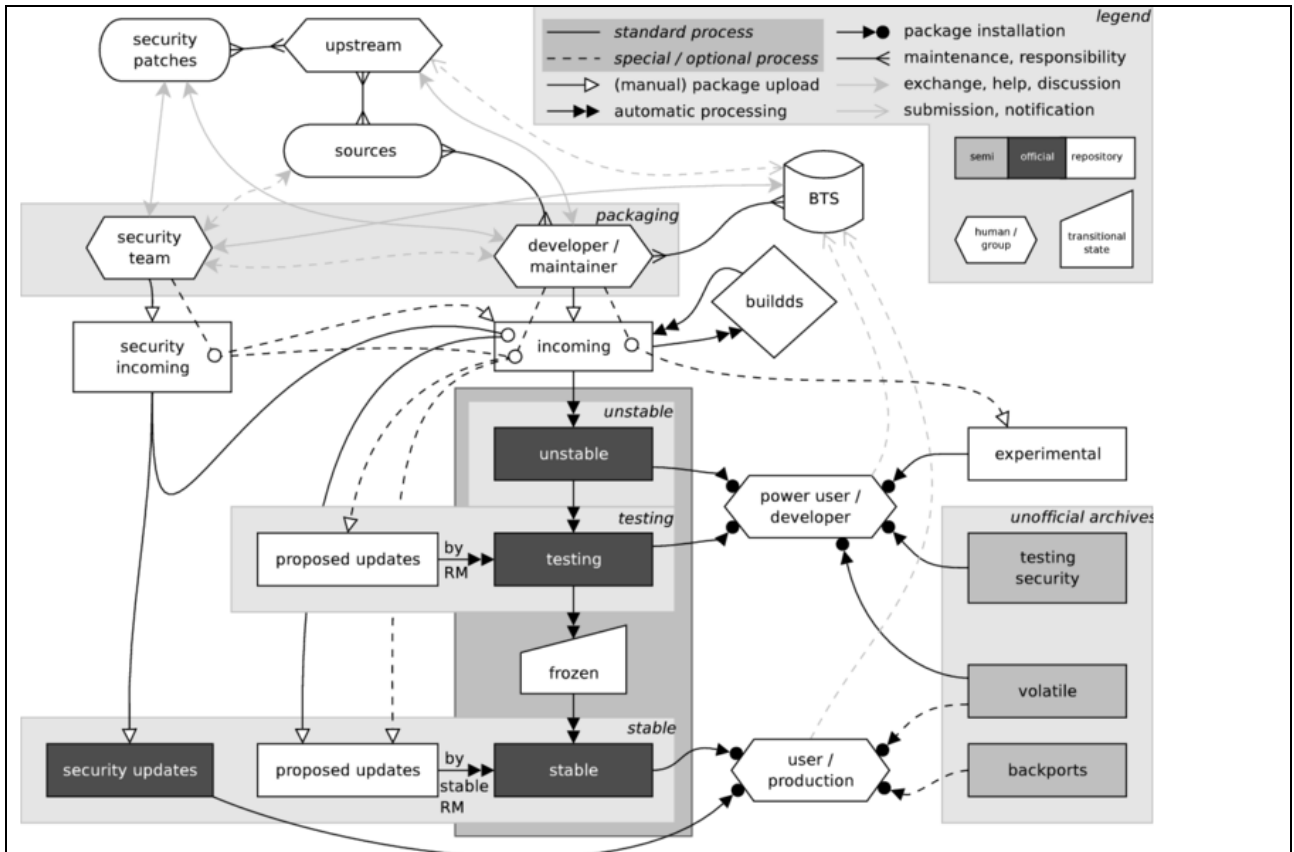
6. References

- (1) Linux Kernel Development: Getting Started by Randy Dunlap Linux Kernel Developer, Mentor, and Janitor. Copyright 2005 Randy Dunlap, All rights reserved, IEEE Northcon May 19, 2005
- (2) Development process document <http://lfn.linuxfoundation.org/book/howparticipate-linux-community>
- (3) Comunidad UBUNTU <http://www.ubuntu.com/project/about-ubuntu/governance>
- (4) Debian Community://www.debian.org/devel/
- (5) Debian Community Organization <http://www.debian.org/intro/organization>
<http://www.debian.org/devel/constitution>
- (6) Debian Processes:
 - a. New members: <http://www.debian.org/devel/join/nm-checklist>
 - b. <http://www.debian.org/devel/join/newmaint#Sponsor>
 - c. <http://www.debian.org/doc/developers-reference/index.html>
 - d. Votes <http://www.debian.org/vote/>
- (7) Ubuntu Community <http://www.ubuntu.com/community>
- (8) Ubuntu Community Organization: <http://www.ubuntu.com/project/about-ubuntu/governance>
- (9) Ubuntu processes: <https://wiki.ubuntu.com/UbuntuTeams>
- (10) Based on Producing Open Source Software: How to Run a Successful Free Software Project by Karl Fogel Copyright © 2005, 2006, 2007, 2008, 2009 Karl Fogel, under a Creative Commons Attribution ShareAlike (3.0) license.

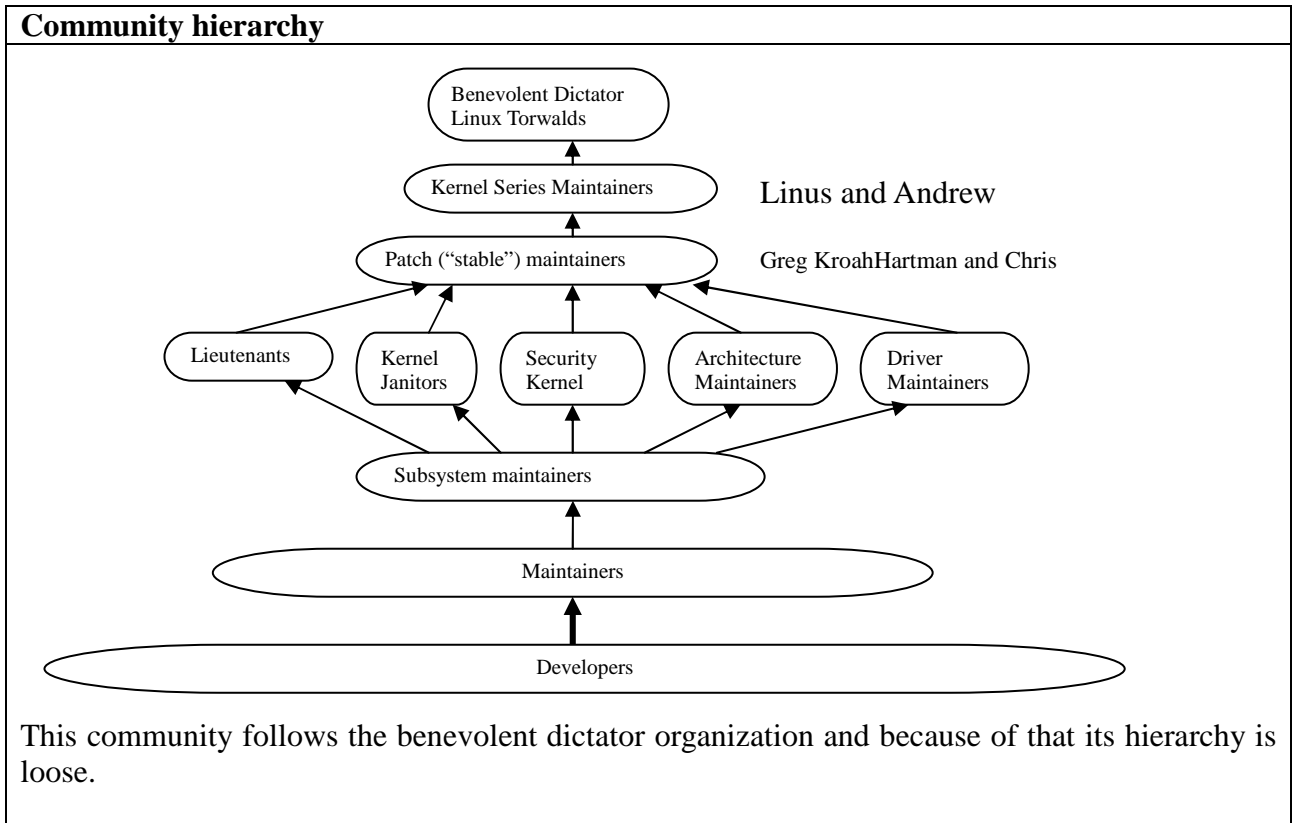
Appendix 1

Name	Debian
Area	O.S.





Name	Linux kernel
Area	kernel



Toplevel maintainers are the ones responsible for integrator and can overrule if it is needed. They take care for the quality of the code. Most of this work is shared and delegate to lieutenants and individual maintainers.

Maintainers don't have absolute authority, only the dictator has that right.

Arch and subsystem maintainers: coordinate subsystems and maintain consistency.

Driver maintainers: cover all current mainline kernels and update to new kernel APIs, even development APIs

Process

New member

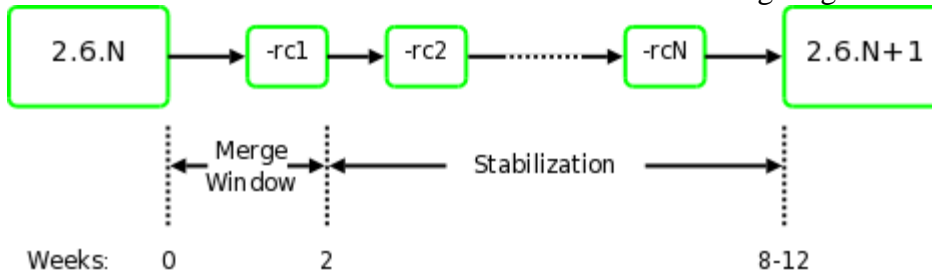
New members are accepted by meritocracy. They should demonstrate their commitment to the project. Developers expect to be maintaining the code 5-10 years from the time is released.

New members should demonstrate how they work by worming with janitor kernel maintainers which are the ones that take responsible for mentoring new developers.

Code release

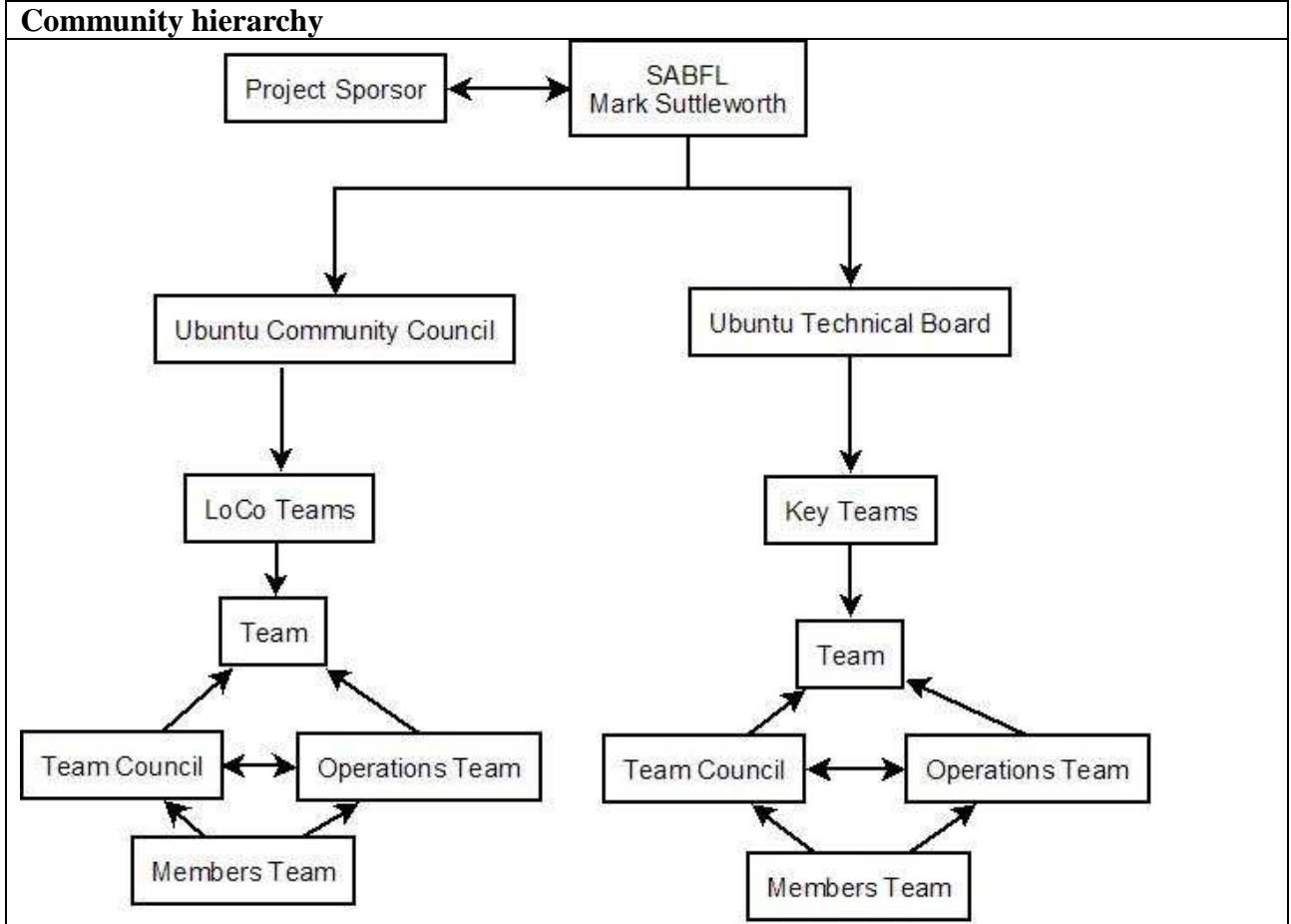
The stable tree is leaded by Linus Torvalds and use a loosely time-based release process, with a 2-3 month release cycle.

The work on the mainline tree is described on the following diagram.

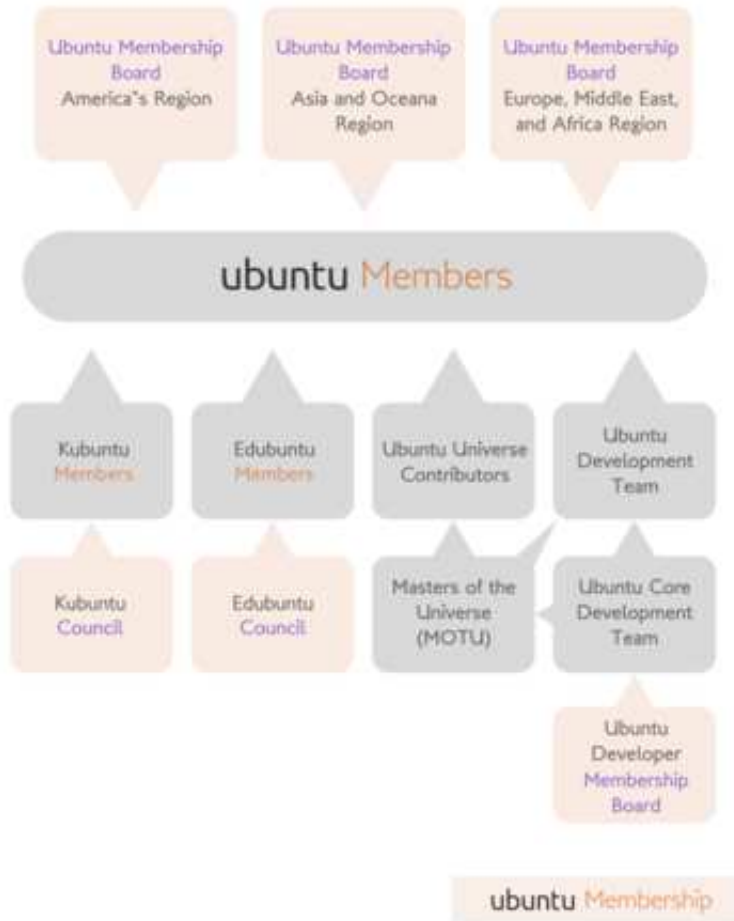


On the next diagram the numbering of the patches are explained

Name	Ubuntu
Members	
Area	O.S.



Process
<i>New member</i>



Applicants should add their name to the agenda for the next meeting of the membership board for your region, or the team council. On this meeting applicants are asked to prove their contribution to Ubuntu.

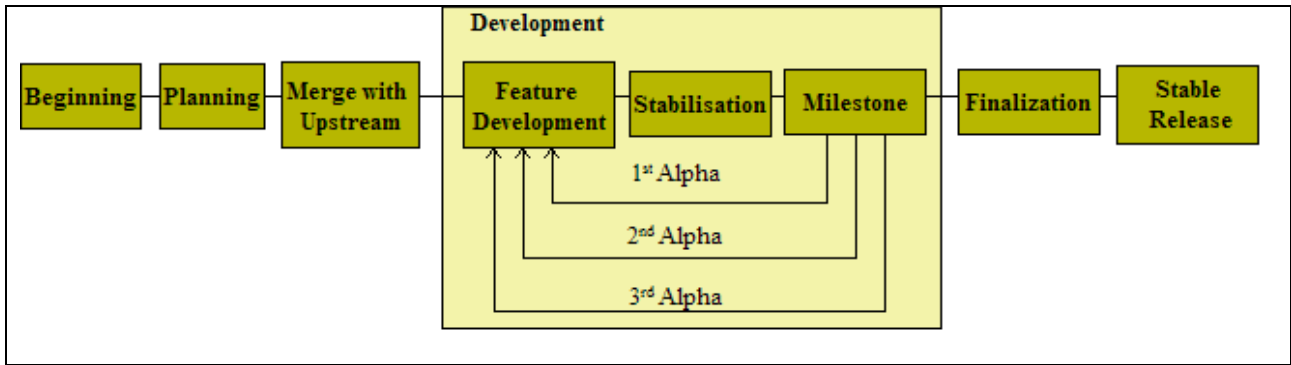
Applicant should have a Personal wiki page with his/her contributions carefully documented on it. The wiki page should include the following details:

1. A summary of his/her contributions to Ubuntu (no longer than 2-3 lines)
2. A link to the applicant's Launchpad profile
3. A complete description of the applicant's contributions to Ubuntu
4. The applicant's plans and ideas for Ubuntu in the near and far future

The applicant must have signed the Code of Conduct, prior to applying for membership.

If there are recognized members of the Ubuntu community supporting the applicants at the meeting, this will definitely speed up the process of approving the membership. If the "sponsors" can't attend the relevant meeting, they can be asked to leave a testimonial on the wiki page about the applicant's contributions.

Code release



Name	Mozilla
Area	Web

Community hierarchy

The diagram shows the community hierarchy. At the top is 'Benevolent Dictators', which has arrows pointing to 'Drivers', 'Brendan Eich Technical issues', and 'Mitchell Baker Non-technical issues'. 'Drivers' has arrows pointing to 'Super-Reviewers', 'Module Owners', and 'Bugzilla Component Owners'. 'Super-Reviewers' has an arrow pointing to 'Module Owners'. 'Module Owners' and 'Bugzilla Component Owners' are connected by a double-headed arrow.

1 The Mozilla project is governed by a virtual management team made up of experts from various parts of the community. Some people with leadership roles are employed to work on Mozilla and others are not. Leadership roles are granted based on how active an individual is within the community as well as the quality and nature of his or her contributions. This meritocracy is a resilient and effective way to guide our global community. The different community leadership roles include:

Module Owners and Peers
[Module owners](#) are responsible for leading the development of a module of code or a community activity. This role requires a range of tasks, including approving patches to be checked into the module or resolving conflicts among community members. Lists of [code module owners](#) and [non-code module owners](#) are available.

Super-Reviewers
[Super-reviewers](#) are a designated group of strong hackers who review code for its effects on the overall state of the tree and adherence to Mozilla coding guidelines. Super-review generally follows code review by the module owner, and the approval of a super-reviewer is generally required to check in code. More information on code review can be found in

the mozilla.org code review FAQ.

Release Drivers

[Release drivers](#) provide project management for milestone releases. The drivers provide guidance to developers as to which bug fixes are important for a given release and also make a range of tree management decisions.

Bugzilla Component Owners

[Bugzilla component owners](#) are the default recipient of bugs filed against that component. Component owners are expected to review bug reports regularly, reassign bugs to correct owners, ensure test cases exist, track the progress toward resolving important fixes, and otherwise manage the bugs in the component. The Bugzilla component owner and the related module owner may be the same person, but in many cases they will be different.

Benevolent Dictators

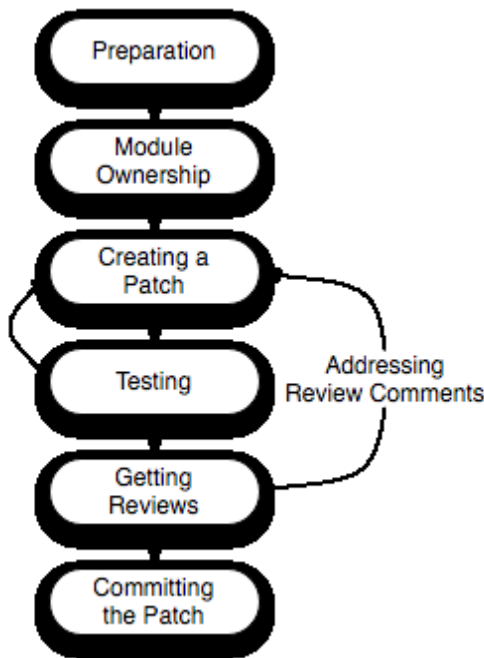
Benevolent dictators are trusted members of the community who have the final say in the case of disputes. This is a [model](#) followed by many successful open source projects, although most of those communities only have one person in this role. Mozilla has evolved to have two people in this role—Brendan Eich has the final say in any technical disputes and Mitchell Baker has the final say in any non-technical dispute.

Process

New member

Code release

Submitting a patch and getting it reviewed and committed to the Mozilla source tree can be a daunting task. There are several steps that should be taken:



One of the main characteristic of Mozilla is the Reviews. Mozilla is the fist community that has created a



SOFIA roadmap: Community Oriented Strategy		28 (28)
D6.41 V1.0	Confidentiality: PU (Public)	2010-08-02

Quality Group.

