ສະຫຼຸບບົດຮຽນ ບົດທີ 6  ການຈັດການຄວາມສ່ຽງ

1. ຄວາມສ່ຽງ ແລະ ການຈັດການຄວາມສ່ຽງ
2. ຄວາມສ່ຽງ

ຄວາມສ່ຽງ ແມ່ນຄວາມເປັນໄປໄດ້ຂອງການເກີດເຫດການທີ່ເຮັດໃຫ້ເກີດຄວາມເສຍຫາຍຕໍ່ໂຄງການທີ່ອາດຈະເປັນໄປໄດ້ຂອງເຫດການດັ່ງກ່າວ.

1. ການຈັດການຄວາມສ່ຽງ

ການຈັດການຄວາມສ່ຽງ ເປັນຂະບວນການທີ່ປະກອບໄປດ້ວຍກິດຈະກຳຕ່າງໆຄື:

1. ບອກຄວາມສ່ຽງທີ່ອາດຈະເກີດຂື້ນ (ກຳນົດປັດໃຈສ່ຽງ).
2. ປະເມີນຫາຄວາມສ່ຽງທີ່ມີຜົນກະທົບຮ້າຍແຮງ.
3. ກະກຽມວາງແຜນເພື່ອລຸດລະດັບຄວາມສ່ຽງຮ້າຍແຮງໃຫ້ສາມາດຄວບຄຸມໄດ້.
4. ກະກຽມປ້ອງກັນບໍ່ໃຫ້ເກີດຄວາມສ່ຽງຕ່າງໆຂື້ນອີກ.

ການຈັດການຄວາມສ່ຽງ:

* Reactive ເປັນການຈັດການກັບຄວາມສ່ຽງໃນທັນທີທີ່ມັນເກີດຂື້ນ.
* Proactive ເປັນການປ້ອງກັນບໍ່ໃຫ້ຄວາມສ່ຽງເກີດຂື້ນ.

1. ປະເພດຂອງຄວາມສ່ຽງ
2. ປະເພດຄວາມສ່ຽງຂອງໂຄງການ (Project Risk)

* ເປັນຄວາມສ່ຽງທີ່ສົ່ງຜົນກະທົບຕໍ່ຕາຕະລາງການເຮັດວຽກຂອງໂຄງການ ແລະ ຊັບພະຍາກອນທີ່ຕ້ອງໃຊ້.

1. Product Risk

* ເປັນຄວາມສ່ຽງທີ່ຜົນກະທົບຕໍ່ຄຸນນະພາບ
* ອາດຈະເກີດຈາກສ່ວນປະກອບຍ່ອຍຂອງຊອບແວຣ໌ບໍ່ມີຄຸນນະພາບ

1. Business Risk

* ເປັນຄວາມສ່ຽງທີ່ສົ່ງຜົນກະທົບຕໍ່ທຸລະກິດຂອງອົງກອນ ແລະ ອາດສົ່ງຜົນກະທົບຕໍ່ຜະລິດຕະພັນ ຫຼື ໂຄງການໄດ້
* ຄູ່ແຂ່ງເອົາຊອບແວຣ໌ອອກວາງຈຳໜ່າຍກອ່ນ, ຜະລິດຊອບແວຣ໌ທີ່ບໍ່ຖືກຕາມຄວາມຕ້ອງການຂອງຕະຫຼາດ.

1. ຂັ້ນຕອນການຈັດການຄວາມສ່ຽງ
2. ການກຳນົດປັດໃຈສ່ຽງ
3. ການວິເຄາະຄວາມສ່ຽງ
4. ການວາງແຜນຄວາມສ່ຽງ
5. ການຕິດຕາມຄວາມສ່ຽງ
6. ການແກ້ບັນຫາຄວາມສ່ຽງ
7. ການກຳນົດປັດໃຈສ່ຽງ

+ ເປັນການຄົ້ນຫາປັດໃຈສ່ຽງທັງໝົດທີ່ອາດຈະເກີດຂື້ນ

+ ຈັດກຸ່ມຂອງປັດໃຈສ່ຽງ

* ປັດໃຈສ່ຽງດ້ານເທັກໂນໂລຍີ່
* ປັດໃຈສ່ຽງດ້ານບຸກຄະລາກອນ
* ປັດໃຈສ່ຽງດ້ານອົງກອນ
* ປັດໃຈສ່ຽງດ້ານເຄື່ອງມື
* ປັດໃຈສ່ຽງດ້ານຄວາມຕ້ອງການ
* ປັດໃຈສ່ຽງດ້ານການປະເມີນ

1. ການວິເຄາະຄວາມສ່ຽງ

* ເປັນການປະມານ ແລະ ປະເມີນ ຄວາມສ່ຽງຈາກລາຍການຄວາມສ່ຽງ
* ກຳນົດ ຄວາມອາດຈະເປັນໄປໄດ້ທີ່ຈະເກີດຄວາມສ່ຽງຂື້ນ

1. ການວາງແຜນຄວາມສ່ຽງ

    + ເປັນຂັ້ນຕອນການພິຈາລະນາຄວາມສ່ຽງແຕ່ລະລາຍການທີ່ໄດ້ຮັບຈາກການຄັດ  ເລືອກມາແລ້ວ ມາກຳນົດວິທີເພື່ອຈັດການ

    + ວິທີຈັດການຄວາມສ່ຽງແບ່ງອອກເປັນ 3 ຊະນິດ:

* ການຍອມຮັບ
* ການປ້ອງກັນ
* ການໂອນຍ້າຍ

1. ການຕິດຕາມຄວາມສ່ຽງ

* ເປັນການຕິດຕາມວ່າຈະເກີດຄວາມສ່ຽງຂື້ນ ຫຼື ບໍ່, ເມື່ອໃດ ແລະ ແນວໃດ, ເມື່ອເກີດແລ້ວມີການປ່ຽນແປງ ຫຼື ບໍ່

1. ການແກ້ໄຂບັນຫາຄວາມສ່ຽງ

* ເປັນຂັ້ນຕອນໃນການກຳຈັດຄວາມສ່ຽງໃຫ້ໝົດໄປ ຫຼື ໃຫ້ເຫຼືອໜ້ອຍທີ່ສຸດ
* ຜົນຈາກການແກ້ໄຂບັນຫາຄວາມສ່ຽງແມ່ນຈະເກີດຄວາມສ່ຽງຂືື້ນ ຫຼື ເກີດຂື້ນແຕ່ລຸດລະດັບຄວາມຮຸນແຮງລົງເປັນລະດັບທີ່ຍອມຮັບໄດ້

ບົດທີ 7 ຄວາມຕ້ອງການດ້ານຊອບແວຣ໌

1. ຄວາມຕ້ອງການດ້ານຊອບແວຣ໌

+ ຄວາມຕ້ອງການຂອງລູກຄ້າ ຫຼື ຜູ້ໃຈເປັນຕົວກຳນົດໜ້າທີ່ການເຮັດວຽກ, ຮູບຮ່າງ, ຄວາມສາມາດ ແລະ ລາຍລະອຽດອື່ນໆ ຂອງລະບົບ ແລະ ຊອບແວຣ໌

+ ຄວາມຕ້ອງການມີ 3 ລະດັບ

* ຄວາມຕ້ອງການຂອງໃຊ້
* ຄວາມຕ້ອງການຂອງລະບົບ
* ຄວາມຕ້ອງການດ້ານຊອບແວຣ໌

1. ປະເພດຂອງຄວາມຕ້ອງການດ້ານຊອບແວຣ໌
2. ຄວາມຕ້ອງການທີ່ເປັນໜ້າທີ່ຫຼັກ (Function Requirement)

* ແມ່ນຄວາມຕ້ອງການໃຫ້ຊອບແວຣ໌ເຮັດວຽກເຫຼົ່ານັ້ນໄດ້ຕາມທີ່ກຳນົດເອົາໄວ້
* ຕົວຢ່າງລະບົບລົງທະບຽນຂອງມະຫາວິທະຍາໄລ ໂດຍມີຜູ້ໃຊ້ແມ່ນ ນັກສຶກສາ, ອາຈານສອນ ແລະ ພະລັງງານລົງທະບຽນທີ່ຕ້ອງການໃຫ້ຊອບແວຣ໌ເຮັດວຽກໄດ້

1. ຄວາມຕ້ອງການທີ່ບໍ່ເປັນໜ້າທີ່ຫຼັັກ (Non-Functional Requirement )

* ເປັນຄວາມຕ້ອງການທີ່ບໍ່ໄດ້ກ່ຽວຂ້ອງໂດຍກົງກັບໜ້າທີ່ຫຼັກຂອງລະບົບ

1. ຄວາມຕ້ອງການທາງດ້ານທຸລະກິດ

* ເປັນຄວາມຕ້ອງການທີ່ກ່ຽວຂ້ອງກັບວຽກງານທາງທຸລະກິດທີ່ຕ້ອງການຊອບແວຣ໌ມາສະໜັບສະໜູນສະເພາະ
* ໂດຍທີ່ສາມາດເປັນໄປໄດ້ທັງຄວາມຕ້ອງການທີ່ເປັນໜ້າທີ່ຫຼັກຂອງລະບົບ,ເປັນເງື່ອນໄຂຕ່າງໆໃຫ້ແກ່ການຄຳນວນຫາຄ່າຕ່າງໆຂອງລະບົບ

1. ຄວາມຕ້ອງການຂອງຜູ້ໃຊ້

* ເປັນຄວາມຕ້ອງການຂອງຜູ້ໃຊ້ທີ່ມີຕໍ່ລະບົບຊຶ່ງກຳນົດໂດຍຜູ້ໃຊ້ລະບົບ ດ້ວຍສ່ວນທີ່ເປັນໜ້າທີ່ຫຼັກ ແລະ ສຳຮອງຂອງຂອງລະບົບ ດ້ວຍພາສາທີ່ຜູ້ໃຊ້ອ່ານເເລ້ວເຂົ້າໃຈໄດ້ງາ່ຍ
* ເປັນຄວາມຕ້ອງການລະດັບສູງ ຊຶ່ງບໍ່ໄດ້ກຳນົດລາຍລະອຽດ ແລະ ເງື່ອນໄຂ
* ຫຼັກການສຳລັບການຂຽນຄວາມຕ້ອງການຂອງຜູ້ໃຊ້
* ກຳນົດມາດຕະຖານຮູບແບບເອກະສານ ເຊັ່ນ: ຕົວອັກສອນ, ຂະໜາດ, ສີຕົວອັກສອນ, ການເນັ້ນຂໍ້ຄວາມ, ການຂີດກ້ອງ, ຕົວເນີ້ງ,...
* ຈຳແນກຄວາມຈຳເປັນຂອງການ ໂດຍແບ່ງອອກເປັນ “ຄວາມຕ້ອງການທີ່ຈຳເປັນ ແລະ ຄວາມຕ້ອງການທີ່ເປັນຄວາມປາຖະໜາ
* ໃນເອກະສານຄວນເນັ້ນຂໍ້ຄວາມທີ່ເປັນປະເດັນສຳຄັນຂອງຄວາມຕ້ອງການ
* ພະຍາຍາມຫຼີກລ້ຽງການໃຊ້ຄຳສັບ ຫຼື ຮູບທາງເທັກນິກ

1. ຄວາມຕ້ອງການດ້ານລະບົບ

* ເປັນການກຳນົດຄວາມຕ້ອງການຂອງການເຮັດວຽກ, ໜ້າທີ່ ແລະ ການບໍລິການຕ່າງໆຂອງລະບົບໃນລາຍລະອຽດ
* ເປັນຄວາມຕ້ອງການທີ່ໄດ້ມາຈາກການວິເຄາະຄວາມຕ້ອງການຂອງຜູ້ໃຊ້(ດ້ວຍຂະບວນການທາງວິສະວະກຳ) ວ່າລະບົບຈະຕ້ອງເຮັດຫັຍງ ແລະ ມີເງື່ອນໄຂແນວໃດແດ່ທີ່ລະບົບຄວນເຮັດ ຫຼື ບໍ່ຄວນເຮັດ
* ການຂຽນຄວາມຕ້ອງການຂອງລ/ະບົບ ຄວນໃຊ້ພາສາທີ່ເຂົ້າໃຈງ່າຍເຊັ່ນ:
* ຂໍ້ກຳນົດການໃຊ້ພາສາໂຄ້ງສ້າງ (Structure Language Specification

**ບົດທີ່ 8 ວິສະວະກຳຄວາມຕ້ອງກັນ**

1 ຄວາມໝາຍຂອງວິສະວະກຳຄວາມຕ້ອງກັນ

ໝາຍເຖິງຂະບວນການທີ່ຈະເຮັດໃຫ້ວິສະວະກອນຊອບແວເຂົ້າໃຈ ແລະ ຮູ້

ແຈ້ງເຖິງຄວາມຕ້ອງການຂອງລູກຄ້າຢ່າງແທ້ຈິງ ດ້ວຍການລວບລວມຄວາມ

ຕ້ອງການ, ກວດສອບ ແລະ ນິຍາມຄວາມຕ້ອງການ ເພື່ອນຳໄປສ້າງເປັນຂໍ້ກຳ

ໝົດຄວາມຕ້ອງການຂອງລະບົບຫຼືຊອບແວຣ໌ທີ່ຈະໃຊ້ເປັນຈຸດເລີ່ມຕົ້ນໃນ

ການພັດທະນາລະບົບ

2 ຂະບວນການວິສະວະກຳຄວາມຕ້ອງກັນ

. ການລວບລວມຄວາມຕ້ອງການ(Requirement Elicitation)

. ການວິເຄາະຄວາມຕ້ອງການ (Requirement Analysis)

. ການກຳໜົດຄວາມຕ້ອງການ (Requirement Specification)

. ການກວດສອບຄວາມຕ້ອງການ (Requirement Validation)

3 ການລອບລວມຄວາມຕ້ອງກັນ

* ວິທີການໃນການລວບລວມຄວາມຕ້ອງການ

1. ການສໍາພາດ

2. ການສະແດງລໍາດັບເຫດການການເຮັດວຽກ

3. ສ້າງຕົ້ນແບບໃຫ້ຜູ້ໃຊ້ເລືອກ

4. ການປະຊຸມເພື່ອຂໍຄວາມຄິດເຫັນ

5. ການສັງເກດ

4 ການວິເຄາະຄວາມຕ້ອງການ

* ເປັນການເອົາຄວາມຕ້ອງການທີ່ລວບລວມໄດ້ມາວິເຄາະ ຫຼື ປະເມີນເພື່ອຈັດ

ກຸ່ມຄວາມຕ້ອງການ ແລ້ວ ຈັດລໍາດັບຄວາມສໍາຄັນ, ເບິ່ງຄວາມຖືກຕ້ອງຊອດ

ຄ່ອງ ແກ້ໄຂຄວາມຄັດແຍ່ງກັນ

* ເອົາຄວາມຕ້ອງການດັ່ງກ່າວໄປສ້າງແບບຈຳລອງ ແລະ ອອກແບບສະຖາປັດຕະຍະກຳເບື້ອງຕົ້ນ ເພື່ອທົດສອບການຍອມຮັບຂອງລູກຄ້າ
* ເມື່ອລູກຄ້າຍອມຮັບໃນຂໍ້ກໍາໜົດຄວາມຕ້ອງການ ຈະໄດ້ ເອກະສານຄວາມຕ້ອງການທັງໝົດ ຊຶ່ງເປັນຂໍ້ຕົກລົງທີ່ທັງສອງຝ່າຍເຫັນດີຮ່ວມກັນ
* ການວິເຄາະຄວາມຕ້ອງການມີຈຸດປະສົງດັ່ງນີ້

1. ເພື່ອຊອກຫາ ແລະ ແກ້ໄຂຄວາມຄັດແຍ່ງລະຫວ່າງຄວາມຕ້ອງການແຕ່ອັນ

2. ເພື່ອຊອກຫາຂອບເຂດຂອງຊອບແວຣ໌ ແລະ ການເຮັດວຽກກັບສະພາບແວດລ້ອມ

3.ເພື່ອສຶກສາຄວາມຕ້ອງການຂອງລະບົບຢ່າງລະອຽດ ເພື່ອໃຊ້ກຳໜົດຄວາມຕ້ອງການນອກລະບົບຂອງຊອບແວຣ໌

6 ການກຳນົດຄວາມຕອນການ

ແມ່ນການສ້າງເອກະສານຄວາມຕ້ອງການເພື່ອສະແດງລາຍລະອຽດຂອງ

ຊອບແວຣ໌ທີ່ສາມາດກວດສອບ, ປະເມີນຄ່າ ແລະ ຍອມຮັບໄດ້ ໄດ້ແກ່ ນິ

ຍາມຂອງລະບົບ, ຄວາມຕອ້ງການຂອງລະບົບ, ຄວາມຕ້ອງການຂອງຊອບ

ແວຣ໌

* ເອກະສານນິຍາມລະບົບ
* ເປັນເອກະສານບັນທຶກຄວາມຕ້ອງການດ້ານລະບົບຂອງຜູ້ໃຊ້
* ສະແດງລາຍການຄວາມຕ້ອງການດ້ານລະບົບຕາມຫລັກການ, ເຫດຜົນ ຫຼື ທີ່ມາຂອງ

ລະບົບ ຊຶ່ງຕ້ອງຊອດຄ່ອງກັບຈຸດປະສົ່ງຂອງລະບົບ

* ສະແດງລາຍລະອຽດສະພາບແວດລ້ອມພາຍນອກລະບົບ
* ຂໍ້ຈໍາກັດ, ຂໍ້ສົມມຸດ ຄວາມຕ້ອງການທີ່ບໍ່ເປັນໜ້າທີ່ຫລັກ
* ອາດຈະມີແບບຈຳລອງຄວາມຕ້ອງການໃນລະດັບສູງຕິດຂັດມານຳ

7 ການກວດສອບຄວາມຕ້ອງກັນ

ເປັນການວິເຄາະ ແລະ ກວດສອບຄວາມຕ້ອງການຄືນວ່າຍັງມີຂໍ້ຜິດພາດ ຫຼື

ບັນຫາທີ່ອາດຈະເກີດຈາກການທັບຊ້ອນຂອງຄວາມຕ້ອງການ

* ການກວດສອບເອກະສານຂໍ້ກຳໜົດຄວາມຕອ້ງການ

1. ກວດສອບຄວາມຖືກຕ້ອງ, ຄວາມທ່ຽງຕົງ, ຄວາມສະເໝີພາບ

2. ກວດສອບຄວາມຊອດຄ່ອງ

3. ກວດສອບຄວາມຄົບຖ້ວນສົມບູນ

4. ກວດສອບຄວາມເປັນໄປໄດ້

5. ສາມາດພິສູດໄດ້

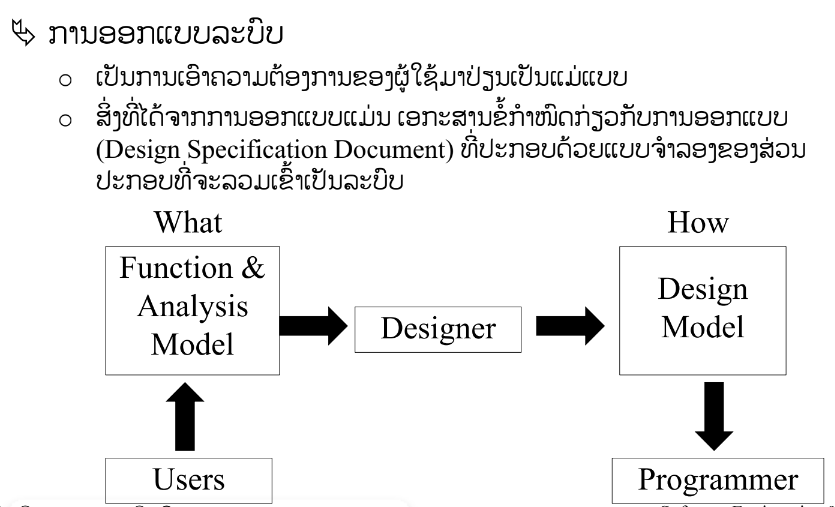
8 ການຈັດການຄວາມຕ້ອງກັນ

ແມ່ນຂະບວນການສ້າງຄວາມເຂົ້າໃຈ ແລະ ຄວບຄຸມການປ່ຽນແປງຄວາມ

ຕ້ອງການຂອງລະບົບ

ບົດທີ່ 9 ການອອກແບບຊວ

1 ຄວາມໝາຍຂອງການອອກແບບຊອບແວຣ໌



* ການອອກແບບຊອບແວຣ໌

ປະສານງານ ແລະ ລັກສະນະອື່ນໆຂອງລະບົບ ຊຶ່ງເປັນການເອົາຄວາມ

ຕ້ອງການຂອງຜູ້ໃຊ້ມາກຳໜົດລາຍລະອຽດໂຄງສ້າງພາຍໃນຂອງຊອບ

ແວຣ໌

ສິ່ງທີ່ໄດ້ຈາກການອອກແບບຄື ແບບຈຳລອງຂອງການອອກແບບ

(Design Model)

2 ຂະບວນການອອກແບບຊອບແວຣ໌

* ຂະບວນການອອກແບບຊອບແວຣ໌ຈະເປັນການເຮັດວຽກແບບຊໍ້າໆ
* ຂະບວນການອອກແບບຊອບແວຣ໌ແບ່ງອອກເປັນ 2 ລະດັບ

- ການອອກແບບສະຖາປັດຕະຍະກຳ

ເປັນການກຳໜົດລັກສະນະໂຄງສ້າງຂອງລະບົບ ຫຼື ຊອບແວຣ໌ ຊຶ່ງສະແດງໃຫ້

ເຫັນສ່ວນປະກອບຕ່າງໆຂອງຊອບແວຣ໌ໃນການເບີ່ງລະດັບເທິງ (ເປັນ

ການສະແດງໃຫ້ເຫັນສ່ວນປະກອບຕ່າງໆຂອງຊອບແວຣ໌)

- ການອອກແບບໃນລາຍລະອຽດ (Detailed Design)

ເປັນການອະທິບາຍລາຍລະອຽດຂອງແຕ່ລະສ່ວນປະກອບຂອງຊອບແວຣ໌ເພື່ອ

ສະດວກໃຫ້ແກ່ການຂຽນໂປຣແກຣມ

3 ສະຖາປັດຕະຍະກຳ ແລະ ໂຄງສ້າງສະຖາປັດຕະຍະກຳຊອບແວຣ໌

* ສະຖາປັດຕະຍະກຳຊອບແວຣ໌ໝາຍເຖິງການອະທິບາຍການພົວພັນລະຫວ່າງ

ລະບົບຍ່ອຍ ແລະ ສ່ວນປະກອບຂອງມັນເພື່ອກຳໜົດໂຄງສ້າງຫຼືລະບົບພາຍ

ໃນຊອບແວຣ໌

* ໂຄງສ້າງສະຖາປັດຕະຍະກຳແມ່ນການກຳໜົດລາຍລະອຽດໃນແຕ່ລະດ້ານຂອງຊອບແວຣ໌ແລ້ວເອົາມາປະກອບເຂົ້າກັນເປັນຊອບແວຣ໌
* ໂຄງສ້າງສະຖາປັດຕະຍະກໍໄດ້ແບ່ງຮູບແບບຂອງສະຖາປັດຕະຍະກຳ(Architecture Style) ອອກເປັນຫລາຍກຸ່ມ ແຕ່ລະກຸ່ມມີລັກສະນະໂຄງສ້າງ ແລະ ຈຸດປະສົງທີ່ແຕກຕ່າງກັນ
* ຮູບແບບຂອງສະຖາປັດຕະຍະກໍາເປັນຂໍ້ບັງຄັບກົດເກັນທາງດ້ານສະຖາປັດຕະຍະກຳທີ່ສ້າງຂື້ນມາເພື່ອຈໍາແນກກຸ່ມຫຼືໝວດໝູ່ຂອງສະຖາປັດຕະຍະກຳຊອບແວຣ໌
* ຮູບແບບສະຖາປັດຕະຍະກຳແບ່ງອອກເປັນ 5 ຮູບແບບ
* ສະຖາປັດຕະຍະກຳແບບໂຄງສ້າງທົ່ວໄປ (Layer, Pipe and Filter, Blackboard)
* ສະຖາປັດຕະຍະກຳລະບົບແບບກະຈາຍ (Client/Server, ThreeTiers, Broker)
* ສະຖາປັດຕະຍະກຳລະບົບແບບ (Model-View-Controller, Presentation-Abstraction-Control)
* ສະຖາປັດຕະຍະກຳລະບົບທີ່ສາມາດດັດແປງໄດ(Micro-Kernel, Reflection)
* ສະຖາປັດຕະຍະກຳລະບົບແບບອື່ນໆ (Batch, Interpreter, ProcessControl, Rule Based)

4 ຄຸນນະພາບ ແລະ ການປະເມີນຄຸນນະພາບການອອກແບບຊອບແວຣ໌

* ຄຸນລັກສະນະຂອງຄຸນນະພາບ(Quality Attribute)
* ການເຮັດວຽກຂອງໂປຣແກຣມ (Functionality)
* ເບິ່ງຈາກ ຈໍານວນ Feature ແລະ ຄວາມສາມາດຂອງໂປຣແກຣມ
* ເບິ່ງຈາກໜ້າທີ່ທົ່ວໄປຂອງໂປຣແກຣມ ແລະ ຄວາມປອດໄພ
* ຄວາມສາມາດໃນການໃຊ້ງານ (Usability)
* ເບິ່ງຈາກການ feedback ຈາກການໃຊ້ງານຂອງຜູ້ໃຊ້ວ່າມັນໃຊ້ງ່າຍບໍ່ ແລະຮຽນຮູ້ງ່າຍບໍ່
* ຄວາມໜ້າເຊື່ອຖື (Reliability)
* ເບິ່ງຈາກການນັບຄວາມຖີ່ ແລະ ຄວາມຮຸນແຮງຂອງຄວາມຜິດພາດທີ່ເກີດຂື້ນ, ຄວາມຖືກຕ້ອງຂອງຜົນຮັບທີ່ໄດ້, ເວລາສະເລ່ຍຂອງຄວາມບໍ່ສໍາເລັດ, ຄວາມສາມາດໃນການກູ້ຄືນລະບົບ ແລະ ຄວາມສາມາດໃນການຄາດການໄດ້ຂອງໂປຣແກຣມ
* ວິເຄາະ ແລະ ປະເມີນຄຸນນະພາບ (Quality Analysis and Evaluation)

ການວິເຄາະ ແລະ ປະເມີນຄຸນນະພາບເປັນກິດຈະກຳທີ່ຊ່ວຍໃຫ້ໝັ້ນໃຈວ່າຊອບແວຣ໌ທີ່ໄດ້ອອກແບບໄວ້ມີຄຸນນະພາບ

**ບົດທີ 10 ການອອກແບບພາກສວນສື່ສານກັບຜູ້ໃຊ້**

1. ພາສວນສື່ສານກັບຜູ້ໃຊ້ ແລະ ຫຼັກການໃນການອອກແບບ

* ແມ່ນການອອກແບບໜ້າຈໍເພື່ອໃຫ້ຜູ້ໃຊ້ສູ້ສື່ສານກັບລະບົບ.
* ການອອກແບບພາກສ່ວນສື່ສານກັບຜູ້ໃຊ້ຄວນຄຳນຶງເຖິງຫຼັກການອອກແບບຕ່າງໆ.
* ຫຼັກການການອອກແບບຂອງ Theo Mandel ປະກອບດ້ວຍຂໍ້ບັງຄັບ 3 ຢ່າງ

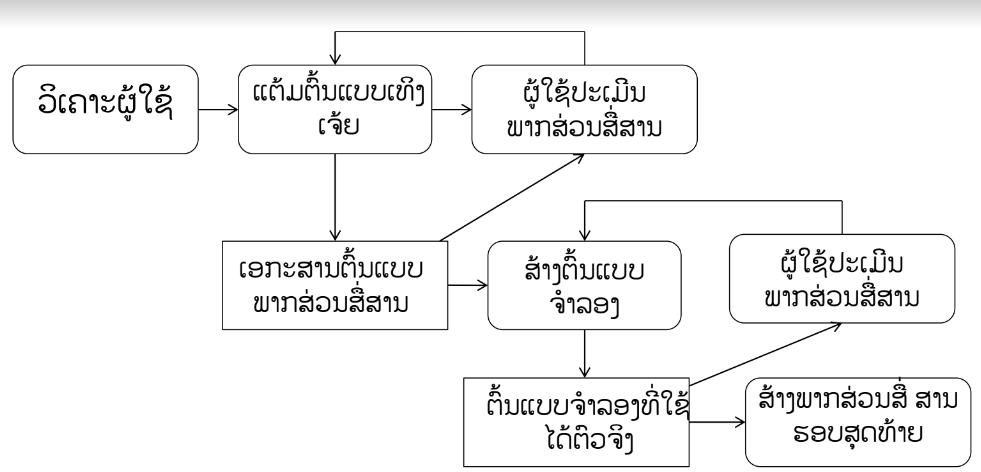
1. ໃຫ້ຜູ້ໃຊ້ສາມາດຄວບຄຸມການເຮັດວຽກບາງຢ່າງໄດ້.
2. ລຸດປະລິມານຂອງສິ່ງທີ່ຜູ້ໃຊ້ຕ້ອງຈື່ຈຳ.
3. ພາກສ່ວນສື່ສານຕ້ອງຊອດຄ່ອງກັນ.

* ໃຫ້ຜູ້ໃຊ້ສາມາດຄວບຄຸມການເຮັດວຽກບາງຢ່າງໄດ້.
* ລຸດປະລິມານຂອງສິ່ງທີ່ຜູ້ໃຊ້ຕ້ອງຈື່ຈຳ.
* ພາກສ່ວນສື່ສານຕ້ອງຊອດຄ່ອງກັນ.

1. ຊະນິດຂອງພາກສ່ວນສື່ສານກັບຜູ້ໃຊ້

* ຊະນິດຂອງພາກສ່ວນສື່ສານກັບຜູ້ໃຊ້ໝາຍເຖິງ ບົດລາຍງານ, ເອກະສານ, ການປ້ອນຂໍ້ມູນ ແລະ ການໂຕ້ຕອບກັບລະບົບ ຊຶ່ງໄດ້ແບ່ງອອກເປັນອອກເປັນ 2 ພາກສ່ວນຄື: ຮູບແບບການໂຕ້ຕອບລະຫ່ວາງຜູ້ໃຊ້ກັບລະບົບ ແລະ ຮູບແບບການນຳສະເໜີຂໍ້ມູນຂ່າວສານ.
* ຮູບແບບການໂຕ້ຕອບລະຫວ່າງຜູ້ໃຊ້ກັບລະບົບ.
* ການໂຕ້ຕອບກັບລະບົບໂດຍກົງ (Direct Manipulation).
* ການເລືອກເມນູຄຳສັ່ງ (Menu Selection).
* ການປ້ອນຂໍ້ມູນລົງໄປໃນຟອມ (Form Fill- In).
* ການໂຕ້ຕອບດ້ວຍພາສາທຳມະຊາດ (Natural Language).
* ການນຳສະເໜີຂໍ້ມູນຂ່າວສານໃຫ້ຜູ້ໃຊ້.

1. ຂະບວນການອອກແບບສ່ວນປະສານກັບຜູ້ໃຊ້

* ປະກອບດ້ວຍກິດຈະກຳຍ່ອຍດັ່ງນີ້
* ການວິເຄາະຜູ້ໃຊ້ (User Analysis).
* ຄວາມຕ້ອງການພາກສ່ວນສື່ສານຂອງຜູ້ໃຊ້.
* ວິເຄາະວຽກທີ່ຜູ້ໃຊ້ເຮັດໃນແຕ່ລະວັນ.
* ວິເຄາະຂໍ້ມູນຂ່າວສານທີ່ຕ້ອງການນຳສະເໜີ.
* ວິເຄາະສະພາບແວດລ້ອມການເຮັດວຽກຂອງຜູ້ໃຊ້.
* ສ້າງຕົ້ນແບບພາກສ່ວນສື່ສານ (Interface Prototyping).
* ຈຸດປະສົງຂອງການສ້າງຕົ້ນແບບກໍ່ເພື່ອໃຫ້ຜູ້ໃຊ້ໄດ້ທົດລອງໃຊ້ລະບົບກ່ອນການໃຊ້ຕົວຈິງ.
* ຂັ້ນຕອນໃນການສ້າງແບບຈຳລອງ ເຊັ່ນ: ແຕ້ມແບບໃສ່ເຈ້ຍແລ້ວໃຫ້ຜູ້ໃຊ້ປະເມີນຕົ້ນແບບນັ້ນ.
* ວິທີ Storyboard ເປັນວິທີໜຶ່ງທີ່ໃຊ້ສ້າງຕົ້ນແບບ ໂດຍການແຕ້ມລັກສະນະຂອງພາກສ່ວນປະສານງານກັບຜູ້ໃຊ້ລົງໃນເຈ້ຍຕາມລຳດັບການໂຕ້ຕອບລະຫ່ວາງຜູ້ໃຊ້ກັບລະບົບເພື່ອໃຫ້ຜູ້ໃຊ້ປະເມີນ.
* ວິທີ Wizard of Oz ເປັນການສ້າງຕົ້ນແບບດ້ວຍການໃຊ້ຊອບແວຣ໌ຈຳລອງເພື່ອໃຫ້ໂຕ້ຕອບກັບຜູ້ໃຊ້ເໜືອນຈິງ
* ການປະເມີນພາກສ່ວນສື່ສານ (Interface Evaluation) ແມ່ນໃຫ້ຜູ້ໃຊ້ປະເມີນຕາມຕົ້ນແບບທີ່ເຮັດຂື້ນມາວ່າຖືກຕ້ອງຕາມຄວາມຕ້ອງການ ຫຼື ບໍ່, ມີຂໍ້ຜິດພາດອັນໃດບໍ່ ໂດຍທີມງານຈະເກັບກຳຄວາມຄິດເຫັນຂອງຜູ້ໃຊ້ໃນການປະເມີນໃຫ້ໄດ້ຫຼາຍທີ່ສຸດເພື່ອນຳໄປປັບປຸງຕົ້ນແບບໃຫ້ສົມບູນ.

**ວິຊາ ວິສະວະກຳຊອບແວຣ໌**

**ບົດທີ 11 ການຂຽນໂປຣແກມ**

1. ມາດຕະຖານການຂຽນໂປຣແກມ

* ເພື່ອໃຫ້ການປະກອບຊອບແວຣ໌, ການເຮັດເອກະສານ ແລະ ການບຳລຸງຮັກສາງາ່ຍຂື້ນຈຳເປັນຈະຕ້ອງມີມາດຕະຖານ ແລະ ຂະບວນການເຮັດວຽກເພື່ອຄວບຄຸມການເຮັດວຽກຂອງໂປຣແກຣມເມີໃຫ້ຊອດຄ່ອງ, ມີລະບຽບ ແລະ ເປັນໄປຕາມທິດທາງດຽວກັນ
* ໂປຣແກຣມເມີທຸກຄົນຈະຕ້ອງຮັບຮູ້ ແລະ ເຂົ້າໃຈໃນມາດຕະຖານ ແລະ ຂະບວນການທຢ່າງຖືກຕ້ອງ ແລະ ກົງກັນ
* ປະໂຫຍດຕໍ່ໂປຣແກຣມເມີ
* ຊ່ວຍໃຫ້ໂປຣແກຣມເມີຈັດລຳດັບຄວາມຄິດໄດ້ຢ່າງເປັນລະບົບ
* ຊວ່ຍໃຫ້ໂປຣແກຣມເມີເຮັດວຽກໄດ້ງາ່ຍຂື້ນ
* ຫຼິກລ້ຽງຂໍ້ຜິດພາດໃນເບື້ອງຕົ້ນ ເຊັ່ນ: ການກຳນົດຊື່ຕົວປ່ຽນ,ຊື່ຖານຂໍ້ມູນທີ່ບໍ່ກົງກັນກັບທີມອືື່ນ
* ປະໂຫຍດຕໍ່ບຸກຄົນອື່ນ
* ເຮັດໃຫ້ການປະກອບ ແລະ ທົດສອບຊອບແວຣ໌ດ້ວຍໂປຣແກຣມເມີຄົນອື່ນງ່າຍຂື້ນ

1. ຫຼັກການໃນການຂຽນໂປຣແກຣມ

* ໂຄງສ້າງການວຄວບຄຸມການເຮັດວຽກຂອງໂປຣແກຣມ
* ການຄວບຄຸມການເຮັດວຽກຂອງໂປຣແກຣມຍ່ອຍຂອງຊອບແວຣ໌ມີຫຼາຍຮູບແບບເຊັ່ນ: Call and Returen, Event-Based Control ແລະ broadcast Model
* ການຄວບຄຸມການເຮັດວຽກຂອງໂປຣແກຣມຍ່ອຍມີຄວາມສຳຄັນຫຼາຍເນື່ອງຈາກມັນເຮັດໃຫ້ໂປຣແກຣມເຮັດວຽກໄດ້ໄປຕາມທິດທາງທີ່ຕ້ອງການໄດ້
* ວິທີການຂຽນໂປຣແກຣມທີ່ຊ່ວຍໃຫ້ການອ່ານໂຄດໃຫ້ງ່າຍນັ້ນມີ 2 ວິທີການຄື:

1. ຂຽນໂຄດຈາກເທິງລົງລຸ່ມ

2 ແບ່ງໂປຣແກຣມອອກເປັນໂມດູນ

1. ການເຮັດເອກະສານກຽ່ວກັບໂປຣແກຣມ

* ການເຮັດເອກະສານພາຍໃນ
* ເປັນເອກະສານສະແດງລາຍລະອຽດຂອງໂຄດທັງໝົດຂອງຊອບແວຣ໌ ແລະ ລວມໝາຍເຫດຂອງໂປຣແກຣມນຳ
* Header Comment Block
* ເປັນການຂຽນໝາຍເຫດໄວ້ໃນສ່ວນຫົວຂອງໂປຣແກຣມ, ເປັນສ່ວນສະແດງທີ່ມາ, ໜ້າທີ່ ແລະ ການໃຊ້ງານໂປຣແກຣມ

**ບົດທີ່ 12 ການທົດສອບຊອບແວຣ໌**

1. ຄວາມຮູ້ເບື້ອງຕົ້ນຂອງການທົດສອບຊອບແວຣ໌

* ເປັນກິດຈະກຳທີ່ເຮັດຂື້ນເພື່ອປະເມີນ ແລະ ປັບປຸງຄຸນນະພາບຂອງຊອບແວຣ໌ ໂດຍການກວດຫາຂໍ້ຜິດພາດ ແລະ ບັນຫາໃຫ້ຖືກຕ້ອງ
* ຈຸດປະສົງຂອງການທົດສອບຊອບແວຣ໌ແມ່ນເພື່ອພິສູດວ່າຊອບແວຣ໌ເຮັດວຽກໄດ້ຄົບທຸກໜ້າທີ່ຕາມຂໍ້ກຳນົດຄວາມຕ້ອງການ ແລະ ແຕ່ລະໜ້າທີ່ສາມາດປະມວນຜົນຂໍ້ມູນໄດ້ຢ່າງຖືກຕ້ອງ
* ຄຳສັບທີ່ຄວນຮູ້ຈັກ
* Error ໝາຍເຖິງຄ່າຈິງທີ່ໄດຈາກການເຮັດວຽກບໍ່ກົງກັບຄ່າຖືກຕ້ອງ ແລະ ລວມເຖິງຜົນການຕັດສິນໃຈຜິດຈາກຄວາມຕ້ອງການ
* Fault ໝາຍເຖິງສະພາບທີ່ຂະບວນການປະມວນຜົນຂອງຊອບແວຣ໌ບໍ່ປົກກະຕິ
* Failure ໝາຍເຖິງຊອບແວຣ໌ ຫຼື ຮາດແວຣ໌ບໍ່ສາມາດເຮັດວຽກຕາມໜ້າທີ່ໃດໜຶ່ງ ລວມເຖິງບໍ່ສາມາດແຈ້ງເຕືອນຂໍ້ຜິດພາດ
* ລະດັບການທົດຊອບແວຣ໌
* ການທົດສອບໃນລະດັບຫົວໜ່ວຍຍ່ອຍ
* ເປັນການທົດສອບແຕ່ລະພາກສ່ວນຍ່ອຍສຸດຂອງຊອບແວຣ໌ ເພື່ອປະເມີນການເຮັດວຽກໃນດ້ານຕ່າງໆ
* ການທົດສອບໃນລະດັບລວມ
* ເປັນການທົດສອບການເຮັດວຽກຂອງກຸ່ມໂປຣແກຣມ ຊຶ່ງເປັນການທົດສອບຫຼັງຈາກທີ່ເອົາແຕ່ລະພາກສ່ວນຍ່ອຍມາລວມເຂົ້າກັນ
* ການທົດສອບລະບົບ
* ເປັນການທົດສອບຊອບແວຣ໌ເມື່ອເອົາມາລວມເຂົ້າກັບອົງປະກອບອື່ນຂອງລະບົບ
* ວິທີທາງໃນການທົດສອບຊອບແວຣ໌
* ວິທີທາງໃນການທົດສອບຊອບແວຣ໌ທີ່ດີທີ່ສຸດແມ່ນການທົດສອບຕາມຮອບຂອງການສ້າງຊອບແວຣ໌ໂດຍເລີ່ມຈາກການທົດສອບເທື່ອລະໂມດູນ ເອີ້ນວິທີການນີ້ວ່າ Incremental Testing Approach ຄືດັ່ງລຸ່ມນີ້:

ອອກແບບກໍລະນີທົດສອບ

Test Case

ກະກຽມຂໍ້ມູນທົດສອບ

ລາຍງານການທົດສອບ

ປຽບທຽບຜົນການສອບ

ທົດສອບໂປຣແກຣມ

Test Data

ຜົນການທົດສອບ

1. ການທົດສອບການລວມຫົວໜ່ວຍຍ່ອຍ

* ເປັນການທົດສອບການເຮັດວຽກຂອງກຸ່ມໂປຣແກຮມ ຫຼື ທົດສອບການລວມໂປຣແກຮມຍ່ອຍເຂົ້າດ້ວຍກັນ ໂດຍເຮັດໜ້າທີ່ໃດໜຶ່ງຮ່ວມກັນ
* ເປັນການທົດສອບພາກສ່ວນສື່ສານການເຮັດວຽກຮ່ວມກັນລະຫ່ວາງແຕ່ລະສ່ວນຍ່ອຍ
* ສ່ວນທີ່ຈະຖືກທົດສອບມີ 2 ຢ່າງຄື: ພາກສ່ວນສື່ສານ ແລະ ຜົນການເຮັດວຽກຂອງພາກສ່ວນລວມ

1. ການທົດສອບລະບົບ

* ເປັນການທົດສອບການເຮັດວຽກຂອງລະບົບເມື່ອເອົາຊອບແວຣ໌ມາລວມເຂົ້າກັບອົງປະກອບອື່ນໆໄດ້ແກ່ ອຸປະກອນ, ບຸກຄະລາກອນ ແລະ ຂໍ້ມູນ
* ການທົດສອບລະບົບແບ່ງອອກເປັນ 2 ລັກລະນະ
* Alpha and Beta Testing
* Runtime Operation Testing
* Alpha and Beta Testing
* Alpha Testing ເປັນການທົດສອບລະບົບໂດຍຜູ້ໃຊ້ຢູ່ສະຖານທີ່ຜະລິດຊອບແວຣ໌ໂດຍຜູ້ໃຊ້ງານພາຍໃຕ້ສະຖານະການຈຳລອງຂື້ນ
* Beta Testing ເປັນການນຳເອົາຊອບແວຣ໌ໄປໃຫ້ຜູ້ໃຊ້ໄດ້ທົດລອງໃຊ້ງານຊອບແວຣ໌ໃນສະຖານທີ່ຈິງດ້ວຍຕົນເອງໂດຍບໍ່ມີທີມງານຄອຍສັງເກດ
* Runtime Operation Testing
* ເປັນການທົດສອບຂະນະທີ່ລະບົບກຳລັງເຮັດວຽກຢູ່
* ສິ່ງທີ່ຕ້ອງທົດສອບມີດັ່ງນີ້
* ທົດສອບການກູ້ຄືນ
* ທົດສອບໃນກໍລະນີຂັບຂັນ (Stress Testing) ເປັນການທົດສອບໃນສະຖານະການບໍ່ປົກກະຕິ ໂດຍລະບົບຈະຕ້ອງເຮັດວຽກຕໍ່ໄປໄດ້
* ທົດສອບສະມັດຕະພາບ
* ທົດສອບການຮັກສາຄວາມປອດໄພ
* ການທົດສອບການເຮັດເອກະສານ

ບົດທີ 13 ການບຳລຸງຮັກສາຊອບແວຣ

1. ການບໍາລຸງຮັກສາຊອບແວຣ໌

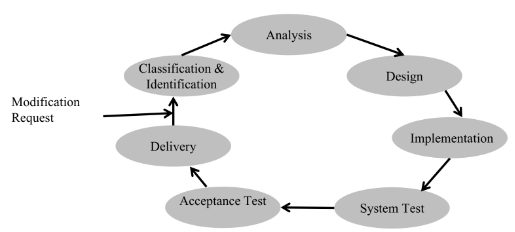
* ແມ່ນການປ່ຽນແປງຊອບແວຣ໌ພາຍຫຼັງການສົ່ງມອບ ເພື່ອແກ້ໄຂຂໍ້ຜິດພາດ, ປັບປຸງປະສິດທິພາບ ຫຼື ດັດແປງຊອບແວຣ໌ໃຫ້ເໝາະສົມກັບສະພາບແວດລ້ອມທີ່ປຽ່ນແປງ ແລະ ຍັງເປັນການປັບປຸງແກ້ໄຂໂຄດ ແລະ ເອກະສານໃຫ້ຖືກຕ້ອງ ເມື່ອມີຂໍ້ຜິດພາດເກີດຂື້ນ ຫຼື ເມື່ອມີການຮ້ອງຂໍໃຫ້ປັບປຸງ
* ຄວາມສຳຄັນຂອງການບຳລຸງຮັກສາ
* ການປ່ຽນແປງເປັນເຫດໃຫ້ມີການບຳລຸງຮັກສາຊອບແວຣ໌
* ເນື່ອງຈາກຊອບແວຣ໌ຕ້ອງເຮັດວຽກຢູ່ພາຍໄຕ້ສະພາບແວດລ້ອມຕ່າງໆທີ່ມີການປ່ຽນແປງຕະຫຼອດເວລາ
* ເພື່ອໃຫ້ແນ່ໃຈວ່າຊອບແວຣ໌ຍັງຄົງເຮັດໄດ້ກົງກັບຄວາມຕ້ອງການຂອງໃຊ້
* ການເລືອກລະຫວ່າງບໍາລຸງຮັກສາ ແລະ ສ້າງຂື້ນໃໝ່
* ເມື່ອມີການປ່ຽນແປງແກ້ໄຂຊອບແວຣ໌ ໝາຍເຖິງອົງກອນຈະຕ້ອງໄດ້ຈ່າຍເງິນຄ່າປັບປຸງສະນັ້ນ ອົງກອນຈະຕ້ອງໄດ້ພິຈາລະນາ ຕັດສິນໃຈເລືອກລະຫວ່າງການບຳຮັກສາ ແລະ ການພັດທະນາຂື້ນໃໝ່ວ່າ ອັນໃດຄຸ້ມຄ່າກ່ວາກັນ
* ປະເພດຂອງການບຳລຸງຮັກສາ
* ການບຳລຸງຮັກສາມີລັກສະນະສຳຄັນ 4 ປະການດັ່ງນີ້
* ຄວບຄຸມການເຮັດວຽກຂອງແຕ່ລະ Function ຂອງຊອບແວຣ໌ໃຫ້ຖືກຕ້ອງເປັນປະຈຳທຸກວັນ
* ຄວບຄຸມການປັບປຸງຊອບແວຣ໌
* ຮັກສາສະພາບຄວາມສົມບູນຂອງ Function ຂອງຊອບແວຣ໌ໄວ້ສະເໜີ
* ປ້ອງກັນບໍ່ໃຫ້ປະສິດທິພາບຂອງຊອບແວຣ໌ລຸດລົງເກີນລະດັບທີ່ສົມຄວນ
* ການບໍາລຸງຮັກສາມີ 4 ປະເພດ
* Corrective Maintenance ເປັນການແກ້ໄຂຂໍ້ຜິດພາດທັນທີທີ່ພົບ
* Adaptive Maintenance ເປັນການດັດແປງສ່ວນທີ່ໄດ້ຮັບຜົນກະທົບ
* Perfective Maintenance ເປັນການປັບປຸງປະສິດທິພາບການເຮັດວຽກ
* Preventive Maintenance ເປັນການປ່ຽນແປງເພື່ອປ້ອງກັນບໍໃຫ້ເຮັດວຽກຜິດພາດ

1. ບັນຫາຂອງການບຳລຸງຮັກສາຊອບແວຣ໌

* ການບຳລຸງຮັກສາບໍ່ໃຫ້ກະທົບຕໍ່ທຸລະກິດຂອງລູກຄ້າ
* ການບຳລຸງຮັກສາຊອບແວຣ໌ເປັນເລື່ອງຍຸ້ງຍາກ
* ຕ້ອງເຮັດໃນຄະນະທີ່ຜູ້ໃຊ້ກຳລັງໃຊ້ງານຢູ່
* ອາດເຮັດໃຫ້ເກີດຄວາມເສຍຫາຍຕໍ່ທຸລະກິດຂອງຜູ້ໃຊ້
* ບໍ່ເຮັດໃຫ້ການເຮັດວຽກຂອງຜູ້ໃຊ້ຢຸດສະງັກ
* ບັນຫາດ້ານເງິນທີມງານ
* ບັນຫາດ້ານເທັກນິກ
* ບັນຫາຈາກການປານີປານອມ
* ບັນຫາດ້ານຕົ້ນທືືນ

1. ຂັ້ນຕອນໃນການບໍາລຸງຮັກສາຊອບແວຣ໌

* ແຜນວາດສະແດງຂັ້ນຕອນໃນການບຳລຸງຮັກສາຊອບແວຣ໌



* ການສະເໜີປ່ຽນແປງແກ້ໄຂ(Modification Request)
* ການຈຳແນກ ແລະ ກຳນົດປະເພດການບຳລຸງຮັກສາ
* ການວິເຄາະ
* ການອອກແບບ
* ການດຳເນີນງານການບໍາລຸງຮັກສາ
* ກາທົດສອບລະບົບ
* ການທົດສອບການຍອມຮັບ
* ການສົ່ງມອບ

1. ເທັກນິກ ແລະ ເຄື່ອງມືໃນການບໍາລຸງຮັກສາຊອບແວຣ໌

* ເທັກນິກໃນການບໍາລຸງຮັກສາຊອບແວຣ໌
* ການສ້າງຄວາມເຂົ້າໃຈໂປຣແກຣມ
* ການຟື້ນຟູສະພາບຂອງຊອບແວຣ໌
* ການປັບປຽ່ນເອກະສານໃໝ່
* ການປັບປ່ຽນໂຄງສ້າງໃໝ່
* ການເຮັດວິສະວະກຳຍ້ອນກັບ
* ການປັບປຽ່ນວິສະວະກຳໃໝ່
* ເຄື່ອງມືໃນການບຳລຸງຮັກສາຊອບແວຣ໌