Nauteff-Vision Conception

Emmanuel Gautier

9 octobre 2024

Résumé

Ce document est le dossier de spécification du logiciel Nauteff-Vision.

Historique du document

Version	Date	Évolutions	réf.
7 novembre 2022	Création du document		
24 juillet 2024	Reprise de la rédaction	Tout	

Table des matières

Chapitre 1

Buts du document

1.1 Buts

C'est le document de référence décrivant la conception du logiciel Nauteff-Vision. Il est issu de sa spécification.

les exigences techniques et opérationnelles et la description des fonctions du logiciel Nauteff-Vision.

Les concepteurs de l'architecture du logiciel et les programmeurs rédigent ce document, les programmeurs s'en servent de référence pour la programmation. Le document de spécification

1.2 Guide de lecture

Le chapitre **??**, page **??** fournit une description générale du logiciel dans son environnement. Le lecteur prenant connaissance du logiciel y trouvera une description générale.

Nauteff-Vision est un complément du Nauteff-AP, la lecture de la spécification de ce dernier est recommandée. Quaternion

Chapitre 2

Documents applicables et de référence

Chapitre 3

Contexte de la conception

3.1 Présentation de l'application

3.2 Principales exigences applicables

3.2.1 Architecture matérielle opérationnelle

Nauteff-Vision fonctionne sur les matériels suivants :
— Nano-ordinateurs avec Linux ou Android ;
— PC portables avec Linux ou Windows ;
— PC de bureau avec Linux ou Windows ;
Nauteff Vision doit pouvoir fonctionner sur des machines peu puissantes et avec peu de ressources.

3.2.2 Exigences de sûreté de fonctionnement

Nauteff Vision est utilisé pour la conduite de navires, il doit avoir les qualités suivantes :
— Fiable ;
— Robuste.

3.2.3 Exigences de facilité de maintenance et d'évolution

Nauteff Vision a vocation à être maintenu et modifié par de nombreux acteurs différents, y compris des utilisateurs ayant peu d'expérience de la programmation. Le code de Nauteff Vision permet de maintenir et faire évoluer le logiciel facilement.

3.2.4 Exigences liées à la performance

Nauteff vision

3.2.5 Logiciels imposés

Néant.

3.2.6 Interface avec d'autres applications

3.3 contraintes de développement

3.3.1 Méthodes et formalisme

3.3.2 Codage

— Documentation très détaillée dans le source du logiciel ;
— Programmation défensive ;
— ...

3.3.3 Langages de programmation et librairies

Le langage de programmation est python 3. Le logiciel utilise les librairies suivantes :
— tkinter pour la partie graphique ;
— threading pour le threads ;
— math, la librairie mathématique ;
— queue pour gérer les files de messages ;
— ios pour accéder aux fichiers et tubes nommés ;

3.3.4 Logiciels acquis ou préexistants

Néant.

3.3.5 Outils

Le développement se fait avec :
— interpréteur python
— eclipse l'EDI au bureau ;
À bord d'un bateau et si l'ordinateur n'a pas assez de puissance ni de mémoire, le développement se fait sans eclipse.

3.3.6 Environnement matériel de développement

Au bureau le développement se fait avec un PC avec des ressources abondantes alors qu'à bord d'un navire le développement se fait sur un PC économe en énergie et donc peu puissant.

Chapitre 4

Architecture générale de l'application

4.1 Architecture statique

4.1.1 Tableau de bord et instruments

Le tableau de bord, désigné Dashboard, contient des instruments et un bandeau de contrôle en bas. Les instruments, désignés InstrumentXxxx, où Xxxx indique le type d'instrument, dérivent de Instrument.

4.1.2 Horloge

L'horloge, désignée tocante, envoie tous les dixièmes de seconde la date, l'heure locale et l'heure UTC au distributeur. Ces dates et heures sont placées sur la queue d'entrée du distributeur. L'horloge fonctionne dans un thread.

4.1.3 Calculs

Les calculs sont fait par des modules de types désignés ComputeXxxx où Xxxx désigne le type de calcul. Ces modules de calcul sont dans un ratachés au distributeur. Chaque module de calcul est exécuté dans thread.

4.1.4 Données et types de données

Chaque donnée est horodatée et a un lien vers un type de donnée. Les données sont produites par le convertisseur de données

4.1.5 Gestionnaire d'historique

4.1.6 Entrées-sorties

Le entrées et sorties sont de fichiers au sens Unix. Ces fichiers peuvent être des périphériques (par ex. /dev/ttyUSB0), des tubes només, voire des fichiers sur disque. DataStream

— Nom ;
— Référence vers la queue du distributeur ;
— Fichier.

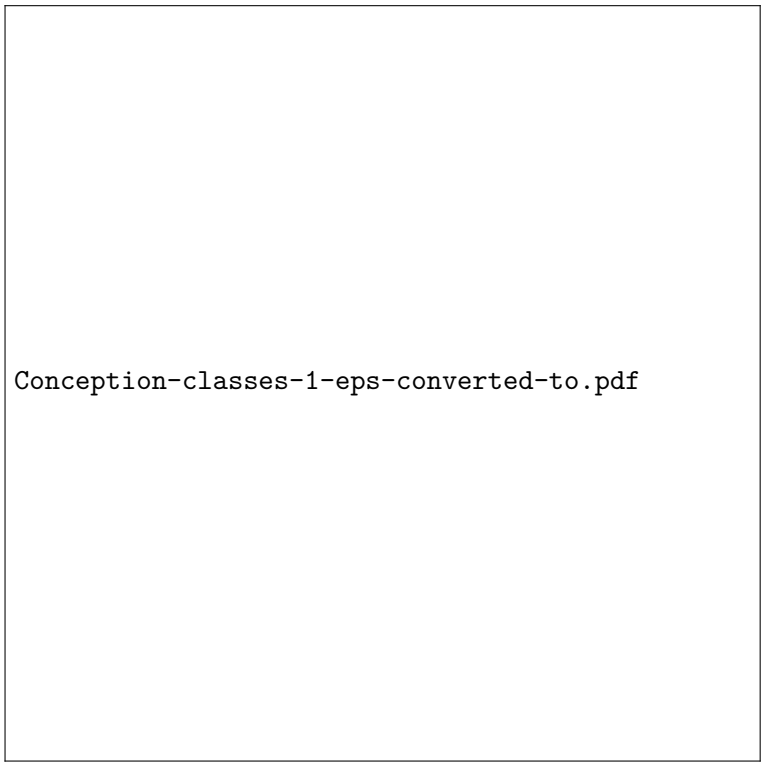
4.1.7 Distributeur

Le distributeur reçoit les données collectées par plusieurs canaux et les redistribue selon leurs types aux récepteurs de données (instruments et modules de calculs) et vers les canaux. Les instruments, les modules de calcul et les entrées sorties indiquent au distributeur la liste des données qu'ils traitent et la liste des données qu'ils émettent.

4.1.8 Simulateur

Le simulateur a vocation à envoyer des données au distributeur lors du développement ou pour des besoins de démonstration. Ce module est en concurrence avec la fonction de relecture.

4.1.9 Relations statiques entre les composants



La classe distributeur contient des canaux d'E/S, un tableau de bord des sorties d'export et des modules de calculs. Le distributeur contient aussi des liens vers les instruments.

4.2 Relations dynamiques

Les échanges entre modules se font par des messages placés dans des queues (FIFO) ou par des fonctions membres.

4.3 Justification des choix d'architecture

La présence de plusieurs entrées et sorties impose l'usage de threads. L'usage de select serait moins souple pour des versions ultérieures avec gestion dynamique des entrées et sorties. tkinter impose de s'exécuter dans son propre thread. Les calculs ont chacun leur thread pour éviter de ralentir ou bloquer l'application en cas de calcul trop long ou infini.

Cette conception modulaire facilite la maintenance et les évolutions et surtout l'ajout de nouveaux calculs ou instruments par dérivation.

Chapitre 5

Architecture des composants

5.1 Dashboard

Le tableau de bord contient des instruments et une barre de contrôle en bas.

5.2 Entrées sorties

5.2.1 Descripteur de fichier

5.2.2 Thread entrée

5.2.3 Thread sortie

Le entrées et sorties sont de fichiers au sens Unix. Ces fichiers peuvent être des périphériques (par ex. /dev/ttyUSB0), des tubes només, voire des fichiers sur disque. DataStream

— Nom ;
— Référence vers la queue du distributeur ;
— Fichier.

Chapitre 6

Description des interfaces