Verification-Centric Realization of Electronic Vote Counting
The original Kiezen op Afstand (KOA) was part of a remote voting system developed for the Dutch government licensed under the GPL. KOA/2 is now a platform for research into computer-based voting and is not intended for use in government elections.
In addition to being Open Source, part of KOA/2 is also formally specified and verified.

The Dutch tally system was formally verified using JML and ESC/Java2.

The Irish vote counting system has been specified using JML and typechecked with ESC/Java2.
class BALLOT
description
   "A ballot paper in an Irish election."
query
   "What is the location of this ballot in the current count?"
   "What is the count number for the last transfer of this ballot?"
   "Is this ballot non-transferable?"
   "What is the first preference of this ballot?"
   "To which candidate is this ballot assigned?"
   "What is the next preference candidate?"
   "What is this ballot's ID number?"
   "Is this ballot paper assigned to a given candidate?"
   "How many preferences remain on this ballot?"
   "Is this ballot on top of a given different ballot?"
command
   "Set this ballot's candidate list."
   "Transfer this ballot to the next preference candidate."
constraint
   "No two ballots have the same ballot ID."
   "A ballot must be assigned to a candidate that is on its candidate list."
   "No two preferences given to candidates on this ballot may be identical."
   "Preferences expressed on this ballot must start with 1, monotonically increase, and grow no larger than the number of candidates available on this ballot."
   "If this ballot is transferred, the count number at which this ballot was last transferred must be positive."
end
JML is a formal behavioral specification language for Java. Annotations embedded in special comments are used to formally express the properties of a Java class/interface. JML is used for modeling of a system or for detailed software specifications using design-by-contract.
Voting systems that we have formally specified differ from country to country.

The Dutch Voting system is list based voters vote for parties, not individuals.

Ireland uses Proportional Representation with a Single Transferable Vote (PRSTV) voters rank individuals by preference.
39 formal assertions were identified in the Count Rules published by the Irish Government.

Each assertion was expressed in JML and identified and cross-referenced by a Javadoc comment.

A state machine was specified so as to link the assertions together.
Section 7 item 3.2 on page 25 of the first source document states:

As a first step, a transfer factor is calculated, viz. the number of votes in the surplus is divided by the total number of transferable votes in the last set of votes. This transfer factor is multiplied in turn by the total number of votes in each sub-set of next available preferences for continuing candidates (note that the transfer factor is not applied to the sub-set of non-transferable votes in the set of votes).

The requirement is written in EBON as follows:

The number of votes in the surplus is divided by the total number of transferable votes in the last set of votes. This transfer factor is multiplied in turn by the total number of votes in each subset of next available preferences for continuing candidates.
/**
 * Determine actual number of votes to transfer to this candidate, excluding
 * rounding up of fractional transfers
 *
 * @see requirement 25 from section 7 item 3.2 on page 25
 *
 * @design The votes in a surplus are transferred in proportion to
 * the number of transfers available throughout the candidates ballot stack.
 * The calculations are made using integer values because there is no concept
 * of fractional votes or fractional transfer of votes, in the existing manual
 * counting system. If not all transferable votes are accounted for the
 * highest remainders for each continuing candidate need to be examined.
 *
 * @param fromCandidate Candidate from which to count the transfers
 * @param toCandidate Continuing candidate eligible to receive votes
 * @return Number of votes to be transfered, excluding fractional transfers
 */

/*@ ensures \result ==
 *  (getSurplus (fromCandidate) *
 *  getPotentialTransfers (fromCandidate, toCandidate.getCandidateID()) /
 *  getTotalTransferableVotes (fromCandidate); */
/**
 * @param {Candidate} fromCandidate - An elected or excluded candidate
 * @param {Candidate} toCandidate - A continuing candidate
 */
/*@ requires fromCandidate.status() != Candidate.CONTINUING; */
/*@ requires toCandidate.status() == Candidate.CONTINUING; */
/*@ ensures fromCandidate.getTotalVote() == */
/*@ old (fromCandidate.getTotalVote() - numberOfVotes); */
/*@ ensures toCandidate.getTotalVote() == */
/*@ old (toCandidate.getTotalVote() + numberOfVotes); */
All the votes recorded at an election in a constituency must be thoroughly mixed together before counting to ensure that vote transfers on the distribution of a surplus are representative.

The votes within each subset of next available preferences for a particular candidate are sorted in the same relative order as the votes were mixed and numbered before the count began.

The particular votes to be transferred from each subset are the votes with the highest random numbers.
KOA/2 remote voting platform used as a case study for verification centric development.

JML and ESCJava/2 tools used to formally specify two different vote counting systems.

Irish vote counting system was specified prior to implementation.

Others writing e-voting systems can look to our work as an example/case study in how "lightweight reliable software engineering" can work.